## ideal

### SİSTEM TANIMLARI PANELİNE ERİŞİM VE PANELİN BÖLÜMLERİ:

Sistem modülüne Grafikler üzerinde bulunan "S" harfine basınca açılan (aşağıda gösterilen) menü üzerinden ulaşır. Modülün en çok kullanılacak kısmı, formül yazmak, yazılmış hazır formüllere ulaşmak, formüllerin çizgi ve diğer parametrelerini değiştirmek gibi işlemlerin yapılacağı SİSTEM TANIMLARI panelidir.



Sistem Tanımları satırına tıkladığınızda ekrana gelen pencere aşağıdaki gibidir.

Sistemler																X
		Kaydet	Yardım	Np	Açıklama	Aktif	Panel	Renk	Kalinlik	Stil		No		Parametre	<u> </u>	
Algo_MA	4	Şifrele	Formul Test	_			1	•	1	1: Düz	-	0				
Algo_MA_veya_R	IZ	Sil	Debug		3		1	÷	1	1:Düz		2	4			
Algo_Tarih_Saat	1	Yazı Tipi	Ondalik 2	_			1	•	1	1:Düz	-	3			L	
Algo_TOMA Algo_YuzdeSean	s	Tindikator Pe	nceresinde Göster				1	Ť	1	1:Duz		5				
Algo_YuzdeSean BARSAYISI	s_Endeks		inceresinae ooster				1	•	1	1:Düz		6				
DEMA5_DEMA21 DEMA5_MAV5			•				1	Ť	1	1:Duz 1:Düz		8				
FilterEhlrs	CLOSE		2				1	٠	1	1:Düz	•	9			-	1
HHV Indicator Mil		🗹 Hata Pencer	esi Compi	ler 0	C Sharp O Visual Basi	с										
Cistem Elementer		I	-	Crofik	Vavilari Vananis 💌	Ortolog			-	usal last	Drice					
, Sisteni Liemaniai	۱ <u>۱</u>		-	Ordrik			ia Taunbu			ijser jeava	nice -		_			_
c																1
													7			
,			6										'			I
M																I
																I
v																

**1 Numaralı Bölge:** Önceden yazılmış ve kaydedilmiş tüm formüller buradan görülebilir. Herhangi bir tanesi seçildiğinde, alt bölgede bu formülün açık kodu, sağ bölümde de varsa çizgi ve parametre değerleri görülür. Grafik üzerinde çalışan bir formül yazılmışsa (indikatör, sistem vs.) doğrudan formülün adını çift tıklamak yeterlidir.

**2 Numaralı Bölge:** Formülü yazdıktan veya var olan formül değişiklik yaptıktan sonra kaydetme, formülde yazım ve mantık hataları var mı diye test etme, kod içine debug satırları konulmuşsa (Bkz debug fonksiyonu) debug sonuçlarını görme, formülü bir başkasına vermeden önce şifreleme işlevleri buradaki butonlarla yapılır.

NOT: ŞİFRELE butonu, kodunuzu üretilen rastgele algoritma ile karıştırıp gizler. ŞİFRELENMİŞ kodu artık hiçbir şekilde geri açamazsınız. Bu nedenle şifrelemeden önce, başka bir isim vererek aynı kodu mutlaka yedekleyin.

**3 Numaralı Bölge:** Formülde ÇİZGİLER varsa, bu çizgilerin panelin bu bölümünden aktif edilmesi gerekir. Kodda kullanılan veya elde edilen bir liste ekrana çizgi olarak çizdirilecekse, her çizgiye bir numara verilir. İLK çizginin numarası SIFIRDIR ve toplam 50 adet çizgi imkânı vardır. Çizgi numaraları kodda **Sistem.Cizgiler[x]** şeklinde yazılır. Sıfır numaralı çizgiyi aktif etmek için panelin bu bölgesindeki kutu işaretlenmelidir.

Her çizginin panel numarası da (koddan da atanabilir) buradan atanır. Grafik zeminin panel numarası 1'dir ve sonraki her alt bölge (indikatör bölgesi) panel numarası artarak gider. Toplam 10 alt panel açılabilir.

Çizgilerin renkleri, kalınlıkları, tipleri bu bölgeden seçilip değiştirilebilir.

**4 Numaralı Bölge:** Formül yazarken, zaman zaman değişen/değişmesi gereken bilgileri kodun içine yazmak yerine, PARAMETRELER isimli bölgede sunulan alanlara yazmak ve formül içinde o bilgiyi kullanmak gerektiği zaman bir komutla o bilgiyi kullanmak işi kolaylaştırır. Grafik periyodu, indikatörün parametreleri, sembol kodu, işlem miktarı, hesap numarası vs. gibi birçok bilgi parametre satırlarına yazılır ve kullanılır.

Kodun içine yazmak yerine Parametreler bölümüne yazıp bir bilgiyi kullanmanın temelde iki nedeni vardır:

- 1- Sıkça değiştirmek durumunda olduğunuz bir bilgiyi kolayca değiştirmek, formül içinde o bilginin yazılı olduğu yeri aramamak.
- 2- Formülü şifreleyip birisine verecekseniz, verdiğiniz kişiye değiştirme imkânı verdiğiniz bilgileri bu alana yazmak. Zira KOD şifrelendiği zaman, parametreler alanı şifrelenmez.

**5 Numaralı Bölge:** Sistem paneli, kodlama yaparken ihtiyaç duyabileceğiniz bazı parametrelerin veya fonksiyonların listelerini size topluca gösterir. "Sistem." Diye başlayarak kullanılan tüm fonksiyonlar Sistem Elemanları kutusunda gösterilir. **Sistem.GrafikVerileri** komutuyla okuduğunuz grafik verileri listesinde hangi alt listeler olduğunu, Hareketli Ortalama (Moving Average) indikatörü için hangi yöntemleri kullanabileceğinizi (ve nasıl yazılmaları gerektiğini), Bir sembole ait YÜZEYSEL fiyatlardan hangilerini kullanabileceğini bu bölgedeki seçim kutularında bulabilirsiniz.

6 Numaralı Bölge: Bu bölge kod/formül yazılan bölgedir.

**7 Numaralı Bölge:** Siz kod yazarken, Sistem. İbaresini yazdıktan sonra, bu bölgede size önerilen/uygun fonksiyonların neler olduğu gösterilir. Gösterilen uygun seçeneklerden birine sağ klik kopyala deyip, kodu yazdığınız yere yapıştır diyerek yazmaya devam edebilirsiniz.

Örneğin Sistem.Grafik dediğinizde, satırı nasıl devam ettirebileceğinizi, hangi ihtimaller olduğunu bu bölge size sunar;



GrafikFiyatOku("IMKBH"YKBNK", "G", "Kapanis") GrafikFiyatOku(GrafikVerileri, "Kapanis") GrafikFiyatSec("Kapanis") GrafikGuncelle(Sembol) GrafikVerileri GrafikVerilerindeTarihHizala(Veriler1, Veriler12) GrafikVerileriniOku(Sembol, "G") GrafikVerisiIndir(Sembol, Periyot)

### SİSTEMLERİN GRAFİĞE UYGULANMASI:

Hazır yazılmış veya sizin yazdığınız bir sistemi grafiğin üzerine uygulamak için yapmanız gereken tek şey, sol bölgedeki sistemin ismine çift tıklamaktır. Bu andan itibaren, formülünüzde var olan çizgiler veya varsa AL/SAT koşullarını gösteren OK'lar, zemine atanmış resim veya yazılar grafik üzerinde gösterilir. Grafik üst kısmındaki S HARFİ yerine, grafiğe atanmış olan sistemin adı gözükür. SİSTEM modülü menülerine girmek için bu isme tıklamak gerekir. Uygulanmış sistemi grafikten kaldırmak için menüdeki ilk satır olan HİÇBİRİ seçeneği tıklanır.

× + VIP	-X030 🖡 Orj 6	0 dk + I T Wal		
	Ağu		Hiçbiri	
Tarih	21.08.2015 17:00	Toplam İşlem 9	Sistem Tanımları	6.80
Açılış Yüksek	90.200 90.750	Kazandıran İşlı Kavhettiren İsl	Performans	1.20
Düşük Kapanıs	90.075 90.600	Reyberter en 19	Son Pozisyonlar	
Hacim	214128500	Karlı İş	Sembol Bazında Sistem Kıyasla	100.000
Ort	90.441	Zararlı Karlı İs	Multi Performans Eğrisi	<u>h.</u>



### SISTEMLERİ ÜZERİNDE DEĞIŞİKLİK YAPMAK VEYA BAŞKA BİR İSİMLE (FARKLI) KAYDETMEK:

Bir sistemin formülünde değişiklikler yaparak, ekrandaki görüntüde (çizgi veya al/sat noktalarında) meydana gelecek değişiklikleri hemen görebilirsiniz.

Bunun için sistemin kodunu açıp (Grafik tepesinde yazan sistem adına basıp, SİSTEM TANIMLARI satırını tıkladığınızda, o sistemin formülü açılır) en alttaki kod kısmında istediğiniz değeri veya parametreyi değiştirip KAYDET butonuna bastığınız anda, yeni değerler anında grafiğe yansır.

ÖRNEK: 27'lik HHV ve 27'lik LLV indikatör değerleri kullanılarak yazılmış aşağıdaki sistem örneğinde, 27 yerine başka bir değer yazıp kaydet butonuna basıldığı anda yeni girilen değere göre hesaplama yapılıp grafikte yeni AL/SAT noktaları gösterilir. Başka başka değerler için sürekli parametreleri girip sistemi çift tıklamadan, değeri değiştirip kaydet dedikçe denemeleri sürdürebilirsiniz.

Bir formülün üzerinde değişiklikler yaptınız ama eskisini bozmayıp, bunu yeni bir isimle kaydetmek istiyorsanız, tek yapmanız gereken, en sol üst köşedeki satıra, sisteme vermek istediğiniz yeni adı girip kaydet butonuna basmaktır.







### SİSTEMLERİ ŞİFRELEYEREK BAŞKALARIYLA PAYLAŞMAK:

Yazdığınız bir formülü/sistemi/robotu başkalarıyla paylaşmak isteyen ama paylaştığı kişinin formülü görmesini istemeyen kullanıcılar ŞİFRELE butonuna basarak kodu gizleyebilirler. Burada dikkat edilmesi gereken çok önemli bir detay vardır. *ŞİFRELENMİŞ BİR SİSTEMİN* <u>KODLARI TEKRAR GÖRÜNÜR YAPILAMAZ.</u> Bu nedenle şifrelemek istediğiniz formülü ÖNCE MUTLAKA BAŞKA BİR İSİMLE kaydedin. Şifrelenmiş formülde, paylaşılan kişinin değiştirmesi gereken veya değiştirmesine izin verilen bilgiler (grafik periyodu, indikatör parametresi, sembol kodu vs) var ise, bu bilgileri kodu içine yazmak yerine PARAMETRELER kutusuna yazarak kullanın. Şifreleme yapıldığı zaman bu alan açık kalacaktır ve şifrelenmiş kod içindeki veriler bu panelden değiştirilebilecektir.

#### SİSTEMLERE KULLANICI BAZINDA LİSANS VE TARİH SINIRI KOYMAK:

izin verdiğiniz kişilerde de çalışmamasını sağlayabilirsiniz.

Yazmış olduğunuz sistem veya robotları, tanıtmak, fikir almak veya başka bir sebeple diğer ideal kullanıcılarıyla da paylaşmak isteyebilirsiniz. Fakat bu paylaşımda eğer formülünüzün/stratejinizin görülmesini istemiyorsanız, formülü mutlaka şifrelemeniz gerekiyor.

Sistem modülündeki şifreleme yapısı, kodun sahibi tarafından atanan bir şifre değeri verilmesi ve sonrasında ancak bu şifrenin girilmesiyle kodun görülmesini sağlayan bir yapı değildir. Burada, şifreleme yapıldığı zaman, iDeal çok fazla parametreyi (o anın zamanı, disk seri numarası ve rastgele üretilen ID vs) kullanarak kodu karıştırır ve görünmez hale getirir. Yani formülün sahibi olan kişi de artık formülün açık halini göremez. Bu nedenle mutlaka, önce başka bir verip kaydetmek ve şifrelemek gerekir.

Şifreleyip bir ideal kullanıcısına verdiğiniz dosyayı, onun da bir başkasına vermesini önlemek isterseniz, formülün sadece sizin belirlediğiniz ideal kullanıcı adıyla login olan kişi/kişiler tarafından görünmesini ve hatta belirlediğiniz bir tarih aralığı içinde çalışıp, o tarihten sonra

Şifreleme: Kodu yazdığınız panel üzerinde bulunan ŞİFRELE butonuna basmak yeterlidir.

Kullanıcı Adına göre lisanslama: Yazılan formülün tamamı EĞER KULLANICI ADI ("X", "Y", "Z") İSE çalış, DEĞİLSE çık şeklinde bir "if" bloğunun içine yazılır.

Kullanıcı adı kontrolü yapan fonksiyonumuz Sistem.LisansKontrol("KullanıcıAdı") şeklinde kullanılır.

Formülün Tarih/Süre Yetkisi Vermek: Yazılan kodun tamamı veya mesela sadece strateji bölgesi (nasıl istenirse) EĞER Tarih1 VE Tarih2 ARALIĞINDAYSA kontrolü içine alınırsa, o formül bu tarihlerin dışındaki zamanlarda çalışmaz.

### ÖRNEK-1: Sadece AHMET, Sezai\_1 ve zeynep2015 kullanıcılarının kullanmasına izin verilen sistem..

```
if (Sistem.LisansKontrol("AHMET","Sezai_1", "zeynep2015"))
{
    var Veriler = Sistem.GrafikFiyatSec("Kapanis");
    var MA1 = Sistem.MA(Veriler, "Exp", 10);
    var MA2 = Sistem.MA(Veriler, "Exp", 100);
    Sistem.KesismeTara(MA1, MA2);
}
```

else

Sistem.Mesaj("Bu Sistemi Kullanmaya Yetkiniz Yoktur");



ÖRNEK-2: Sadece 2015 yılı boyunca çalışsın, 31 Aralık 2015 tarihinden sonra, kullanıcı mesaj ile sürenin bittiği uyarısı verip çalışmasın

```
if (Sistem.TarihAraligi("2015.01.01", "2015.12.31"))
ł
   var TOMA = Sistem.TOMA(3, 2.48);
   var Veriler = Sistem.GrafikFiyatSec("Kapanis");
   var EMA = Sistem.MA(Veriler, "Exp", 3);
   Sistem.KesismeTara(EMA, TOMA);
   Sistem.DolguEkle(1,0,Color.Blue,Color.Pink);
   Sistem.GetiriHesapla("01/01/2010", 0.00);
   Sistem.Cizgiler[0].Deger = Sistem.GetiriKZ;
   Sistem.Cizgiler[1].Deger = Sistem.GetiriKZPoz;
   Sistem.Cizgiler[2].Deger = Sistem.GetiriKZGun;
   Sistem.Cizgiler[3].Deger = Sistem.GetiriKZGun;
   Sistem.Cizgiler[4].Deger = Sistem.GetiriKZYil;
   Sistem.Cizgiler[5].Deger = Sistem.GetiriMiktar;
   Sistem.Cizgiler[6].Deger = Sistem.GetiriPozisyon;
}
else
   Sistem.Mesaj("Sistemin Kullanım Süresi Dolmuştur. abc@xyz.com adresine mail gönderin");
```

:	× ← VIP-X030 -↓ Orj	1dk + I T TOMA →	_ 🗆 🛪
ŀ			
	Tarih 22.10.2015 09:23 Açılış 98.300 Yüksek 98.300	յու հերհերհերհերին հերհերին հերհերին հերհերին հերհերին հերհերին հերհերին հերհերին հերհերին հերհերին հերհերին հ	99.000
	Düşük 98.300	Harthalastarterel Harthalastarterel	98.500
	Kapanış 98.300 Hacim 481670		98.300
	Lot 199 49 Ort 98.300 49	ellar hanserettering and the first and the f	98.000
	🖳 iDeal M	Mesaj 22.10.2015 09:23:06	
	Sistemin	Kullanım Süresi Dolmuştur. abc@xyz.com adresine mail gönderin	

### SISTEMLERIN ISIMLENDIRILMESI:

Yazılan Sistem, Sorgu, Robot, Özel indikatör ve her türlü formül bir isim verilerek kaydedilir. Sistemlere isim verirken aşağıdaki kurallara dikkat edilmelidir:

- a- Sistem isimleri rakamla başlayamaz.
- b- İsimlerde boşluk kullanılmamalıdır. (deneme sistemi gibi bir isim)
- c- Sistem isimlerinde özel işaretler kullanılamaz. (+/-?%&#'! gibi işaretler ve parantezler)
- d- Sistem isimlerinin içinde yada başında alt tire (\_) işareti kullanılabilir.
- e- Sistem isimlerinde Türkçe karakterler (İı, Şş, Ğğ, Üü,Çç,Öö) kullanılamaz.

### FORMÜLLERİN KULLANIM AMAÇLARI (SORGU-SİSTEM-ROBOT-OPTİMİZASYON):

IDEAL sistem modülünde yazılan formüller, çok çeşitli amaçlar için yazılmış olabilir. Mail atmak, alarm veya uyarı amaçlı ses/mesaj çıkarmak, bilgisayara çeşitli verileri yazıp saklamak, ekranın görüntüsünü kaydetmek, grafikler üzerinde çizgi/sinyal/resim/yazı göstermek, otomatik olarak borsaya emir iletmek, sinyal üretip, üretilen sinyalleri önceden girilmiş bir emire bağlamak vs.

Bu işlemleri arttırmak, kullanıcıların amaçlarına ve kodlama yapabilme becerilerine bağlı olarak daha da artabilir. Ama temel olarak IDEALDE bir formülün üç kullanım şekli vardır. Kabaca bir formül bir SİSTEM, bir SORGU, OPTİMİZASYON veya bir ROBOT olabilir. Sistem menüsü altındaki araçlar ve kullanım detayları aşağıda özetlenmiştir.

**SİSTEM:** Yazılan formül grafik üzerine uygulanıyorsa, grafik üzerinde çizgi/yazı/resim vs göstermek veya AL/SAT sinyal noktalarını göstermek, grafik rengini de girilen pozisyona göre belirlemek ve yazılan formülün stratejisine göre Kar/Zarar hesabı, geçmişe dönük performans testleri, getiri kıyaslama yapmak gibi amaçlar için kullanılacaksa bu bir SİSTEMDİR.

**ROBOT:** Sistemler gibi, sizin belirlediğiniz stratejilerde AL veya SAT sinyali üretilen ama diğerlerinden farklı olarak, sinyal üretildiği anda belirlenen sembol için, belirlenen kriterlerde emri doğrudan borsaya ileten formüller ROBOT olarak çalışır. ROBOTLAR, yazılan formülleri, PORFÖY penceresindeki ROBOT sekmesinden, kullanıcı tarafından seçilen sayıda robotu aynı anda sürekli olarak çalıştırır ve formüllerinde yazılı senaryoları işletirler.

En temel kullanım alanı, bir stratejinin gerçekleşmesi durumunda, bir senet veya vadeli sözleşmeye AL veya SAT emrini direk yollamak olmakla birlikte, sürekli çalışan bir koda yaptırılabilecek her türlü iş, ROBOT olarak kaydedilen ve çalıştırılan formüllere yazılabilir. (Her saat xx değerleri mail olarak gönder gibi.)

IDEAL PORTFÖY penceresi açılır. ROBOT sekmesine geçilir. Sol bölgedeki SİSTEM SEÇ butonuna basılır. Kullanıcı tarafından yazılmış tüm formüllerin listesi gelir. Bu listeden çalıştırılacak robot veya robotlar işaretlenir ve kaydet denir. AKSİYON kısmından SANAL seçildiği anda robot veya robotlar çalışmaya başlar. (GERÇEK seçilirse ve bir hesaba login olunmuş ise, sinyal üretildiği zaman gönderilecek emir, gerçek hesaba gönderilir.) Aynı anda çok sayıda robot çalışabilir.

Portföy	Emirler Bekleyen	K/Z Ek	stre Sepet	Portföy	Emirler	Maliyet	Nakit	K/Z	Teyid	Poz.	G.Nakdi	ALGO	ROBOT			
Robot	Yeni_333		Sistem	Seç	Aksiyon	: © Y	ok 🎯	Sanal	🔘 Gerçe	ek	Emir	ler 🔘	Pozisyon	Kontrol		
	Tanımlar 🔄 Bip															
	Uyarı, Yardım Robot_Yeni_333 12:36:24															
No	Sistem		Tar	ih	Aksiyon	S	embol		İşlem	Mikta	r Fiyat	Fij	yat Tipi	Süre	Emir Tipi	Stop
10	Robot_Yeni_333		2015.10.19	11:50:01	Sanal	F_XU030	101550		SATIS	6			PYS	GUN	KPY	

Grafik üzerinde AL/SAT sinyalleri üreten bir sisteminiz varsa ise, bunu aşağıdaki gibi bir ROBOT KALIP kodu kullanarak otomatik emir gönderimine bağlayabilirsiniz.

Bu robot kalıp koduna bir isim verip kaydetmeniz ve sadece ilk 5 satırını kendinize göre değiştirip (sisteminizin adı, grafik periyodunuz, işlem yapacağınız sembol ve işlem adedi bilgileri girerek) kullanabilirsiniz;

```
var LotSize = 1; //işlem adedi
var SistemAdi = "xxxx"; //sistemizin adı
var GrafikSembolu = "IMKBH'DOHOL"; //sistemin sinyal ürettiği grafik sembolü
var GrafikPeriyodu = "60"; //grafiğin periyodu
var EmirSembol = "IMKBH'DOHOL";
var MySistem = Sistem.SistemGetir(SistemAdi, GrafikSembolu , GrafikPeriyodu ); //sistemin adı,
grafik sembolü, grafiğin periyodu
var SonFiyat = Sistem.SonFiyat(EmirSembol);
var Anahtar = Sistem.Name + "," + EmirSembol;
double IslemFiyat = 0;
DateTime IslemTarih:
var Miktar = 0.0;
var Rezerv = "":
var Pozisyon = Sistem.PozisyonKontrolOku(Anahtar, out IslemFiyat, out IslemTarih);
var SonYon = Sistem.SonYonGetir(SistemAdi, GrafikSembolu, GrafikPeriyodu); //sistemin adı, grafik
sembolü, grafiğin periyodu
if (Sistem.Saat.CompareTo("10:00:00") <= 0 || Sistem.Saat.CompareTo("17:59:59") >= 0) // seans
yok işlem yapma
{
}
else
{
   if (SonYon == "F" && Pozisyon != 0) // Flata Gec
      Miktar = -Pozisyon;
   else if (SonYon == "A" && Pozisyon != LotSize) // Al
     Miktar = LotSize - Pozisyon;
   else if (SonYon == "S" && Pozisyon != -LotSize) // Sat
      Miktar = -LotSize - Pozisyon;
    // Emir Gönder
    var Islem = ""
    if (Miktar > 0) {Islem = "ALIS"; Rezerv = "ALIS YAPILDI";}
    if (Miktar < 0) {Islem = "SATIS"; Rezerv = "SATIŞ YAPILDI";}</pre>
    if (Islem != "")
    {
```

```
Sistem.PozisyonKontrolGuncelle(Anahtar, Miktar + Pozisyon, SonFiyat, Rezerv);
Sistem.EmirSembol = EmirSembol ;
Sistem.EmirIslem = Islem;
Sistem.EmirSuresi = "KIE";
Sistem.EmirTipi = "Piyasa";
Sistem.EmirMiktari = Math.Abs(Miktar);
Sistem.EmirGonder();
}
```

**SORGU**: Belli kriterlere uyan hisse veya sembol taratmak için SORGULAR kullanılır. Yatırımcıların en çok ihtiyaç duyabileceği çalışmalardan biri de, kendi belirledikleri kriterlere göre sembol aramaktır. Pek çok strateji veya filtre kriterini sistem modülünde formül olarak yazıp, bu filtreye uyan senetleri bul çalışmasını SORGU analiz aracı üzerinden yapabilirsiniz. Sorgu filtrelerinizde kullanacağınız kriterler/koşullar neredeyse düşünebildiklerinizle sınırlı. Çünkü iDeal'de var olan yüzeysel, derinlik, bilanço, grafik ve indikatör verilerinin tamamını kullanabileceğiniz özel filtreler yazabilirsiniz.

### Örneğin;

}

- Hacmi son 1 aylık hacim ortalamasının üzerine çıkan senetleri bul,
- RSI değeri 70'İ geçen, Fiyatı 100'lük ortalamasının üstünde olan, FK oranı 1 den küçük senetleri bul,
- Derinlikte alış tarafında bekleyen lot miktarı, satış tarafında bekleyen lot miktarının iki katından fazla olan senetleri bul,
- XU100 yüzde getirisi pozitif iken, kendi yüzde getirisi negatif olan senetleri bul,
- Haftalık grafikte Bolinger AL vermiş, günlük grafikte Bolinger orta çizginin altında olanları bul,

Gibi, böyle ve daha kapsamlı binlerce filtre düşünülebilir. Bu örnekleri, hangi verilerle nasıl kriterler yazabileceğinizi göstermek adına örnekledik.

- Sorgu Penceresinde grafik periyotlarından istediğiniz kadar seçebilir ve tarama sonuçlarını aynı anda birden fazla periyot için dökebilirsiniz.
- Taramayı daha hızlı yapabilmek için Son 5000 bar (veya daha az) grafik verisi kullanabilirsiniz.
- Sorgu sonuçlarını kod içinde ilgili komutları yazarak yazı ve zemin renkleriyle işaretleyebilirsiniz.
- Sorgu kriterinize uyan senetler aksini belirtmedikçe en son bardaki değerlere göre filtrelenir. Siz dilerseniz belli bir tarihteki durumu sorgulayabilirsiniz. Tarih kutusunu işaretleyip taramanın sonuçlarının dökülmesini istediğiniz tarihi seçebilirsiniz.
- Sorgu sonuç tablosundaki verileri Excele aktarabilirsiniz.
- Sorgu kriterinin, grafik verisi veya indikatör kullanan bir formül ise, grafik için kullanılacak periyodu seçebilir ve yeni periyot için sorgu sonuçlarını görebilirsiniz.
- Sorgu sonuç tablosunu ekranda açık tutup, istediğiniz zaman aralıklarla otomatik güncellenmesini sağlayabilirsiniz.
- Sorgu formülünü FORMÜL butonuyla açıp, formülde herhangi bir değişiklik yapıp kaydedebilir ve HESAPLA butonuna basarak yapılan değişiklik sonrası yeni sorgu sonuçlarını görebilirsiniz.

- Sorgu analizinin tarayacağı senet/sembol grubunu seçebilir, hazır gruplar kullanabilir veya kendi yarattığınız sembol listesinde sorgulama yapabilirsiniz. (kendi özel sembol listenizi yaratmayı bilmiyorsanız teknik servisimizden destek alabilirsiniz)
- Menü butonuna basarak başka bazı ek ayarlar yapabilirsiniz;
   (örneğin tarama sonucunda çıkan sembolleri otomatik olarak grafik döngü dosyasına kaydetmek veya o senetlerin yazılı olduğu bir fiyat penceresi açtırmak gibi)



• Var olan filtre kriterinizi formüle dökmek konusunda destek almak için <u>algo@idealdata.com.tr</u> adresine mail atabilirsiniz.

Örnek: RSI için sorgu/tarama yapmak ve aşırı alım/satım bölgesinde olan senetleri sorgula.

```
Sistem.SorguBaslik[0] = "Acilis";
Sistem.SorguBaslik[1] = "Yüksek";
Sistem.SorguBaslik[2] = "Düşük"
Sistem.SorguBaslik[3] = "Kapanış";
Sistem.SorguBaslik[4] = "Hacim";
Sistem.SorguBaslik[6] = "RSI 14";
Sistem.SorguBaslik[7] = "MOM 12";
var RSI = Sistem.RSI(14);
var MOM = Sistem.Momentum(12);
var SonRSI = RSI[Sistem.BarSayisi-1];
var SonMOM = MOM[Sistem.BarSayisi-1];
if (SonRSI < 35 || SonRSI > 65)
{
   Sistem.SorguDeger[0] = Sistem.GrafikVerileri[Sistem.BarSayisi-1].Open;
   Sistem.SorguDeger[1] = Sistem.GrafikVerileri[Sistem.BarSayisi-1].High;
   Sistem.SorguDeger[2] = Sistem.GrafikVerileri[Sistem.BarSayisi-1].Low;
   Sistem.SorguDeger[3] = Sistem.GrafikVerileri[Sistem.BarSayisi-1].Close;
   Sistem.SorguDeger[4] = Sistem.GrafikVerileri[Sistem.BarSayisi-1].Vol;
   Sistem.SorguDeger[6] = SonRSI;
   Sistem.SorguDeger[7] = SonMOM;
   if (SonRSI < 30.0F)
      Sistem.SorguAciklama = "Aşırı Satım";
   else if (SonRSI > 70.0F)
      Sistem.SorguAciklama = "Aşırı Alım";
   else
      Sistem.SorguAciklama = "Normal";
   Sistem.SorguEkle();
}
```

## ideal

Yukarıdaki formülü kaydedip, S menüsü altındaki SORGU penceresini açıp, taramanın yapılmasını istediğimiz periyotları seçip, her bir periyotta taratılan senet/sembol grupları için sonuçları aşağıdaki gibi ekrana dönebilirsiniz.

Sisten	Sistem Sorgu													
Sorgu	1		▼ XU-	100 Fo	ormul Menu	🔲 Güncelle	900 saniy	/e Hesapla	Dur	Bar Sayısı	İnceleme	e Ta		
5	Son 5 55 105 155 V 1 2 4 V 5 8 10 15 20 30 60 120 240 5 V G H A O Tüm Barları											3ar		
No	Sembol	Periyot	Açıklama	Açılış	Yüksek	Düşük	Kapanış	Hacim		RSI 14	MOM 12			
1	ARCLK	1	Normal	15.5200	15.5200	15.5200	15.5200	1408099		68.2649	100.1290			
2	ASELS	G	Normal	29.4600	29.6200	29.1600	29.2800	576816000		65.2789	108.2040			
3	BIMAS	5	Aşırı Alım	63.2000	63.2000	63.2000	63.2000	5417631		75.6624	101.6077			
4	BIMAS	G	Aşırı Alım	61.9500	64.3000	61.4500	63.2000	249476700		71.9571	122.4806			
5	EKGYO	G	Normal	1.3500	1.3700	1.3400	1.3600	112110100		66.1677	110.5691			
6	GARAN	1	Normal	7.5400	7.5400	7.5400	7.5400	12098590		32.8429	99.8676			
7	GARAN	G	Normal	7.5600	7.6000	7.5100	7.5400	496959300		34.8465	92.6290			
8	KOZAA	1	Aşırı Satım	12.4600	12.4600	12.4600	12.4600	6528830		23.1357	99.3621			
9	KOZAA	5	Aşırı Satım	12.4600	12.4600	12.4600	12.4600	6528830		23.5282	98.4980			
10	KOZAL	1	Aşırı Satım	73.6000	73.6000	73.6000	73.6000	3406724		24.7802	99.6615			
11	KOZAL	5	Aşırı Satım	73.6000	73.6000	73.6000	73.6000	3406724		21.3409	98.9913			
12	TAVHL	1	Aşırı Alım	18.7100	18.7100	18.7100	18.7100	1372790		79.3470	100.1070			
13	TAVHL	5	Aşırı Alım	18.7100	18.7100	18.7100	18.7100	1372790		78.8149	101.2446			
14	TCELL	1	Normal	14.1500	14.1500	14.1500	14.1500	2898458		32.0460	99.7181			
15	TCELL	5	Normal	14.1500	14.1500	14.1500	14.1500	2898458		30.4962	99.0896			

**OPTIMIZASYON:** Sistem modülü altındaki analiz araçları, yatırımcıların kafalarındaki bazı sorulara cevap bulmak amacıyla kullanılmaktadır. Optimizasyon analiz aracı da bu sorular arasından önemli birine cevap bulmak için geliştirilmiştir: AL/SAT Sinyalleri üreten formülünüzdeki stratejinizde kullandığınız değerler veya indikatör parametreleri ne olursa sisteminiz en iyi getiriyi sağlar?

Örneğin, iki hareketli ortalamanın birbirini kesmesiyle AL/SAT sinyalleri üreten bir sistem kullanıyorsunuz, ama bu *hareketli ortalamaların periyotları ne olmalı?* 

Veya RSI indikatörünün, belli bir değeri yukarı kesmesiyle AL yapacaksınız, ama <u>RSI HANGİ</u> <u>DEĞERİ YUKARI KESERSE?</u>

İşte bu sorulara cevap bulmak için, stratejinizi içeren sistemi yazdıktan sonra, Optimizasyon Analiz aracından geçirecek birkaç ilave yapıp, sonuçları görebilir ve sonra, bulduğunuz ve uygun gördüğünüz sonuçları gidip sisteminizde değiştirip kullanmaya başlayabilirsiniz.

- Yazılı bir sistemi optimize edebilmek için, Optimizasyon analizi için gerekli birkaç değişiklik yaparak sistemi Optimizasyon Formülü olarak da kaydedin.
- Sisteminizde, en uygun değerini aradığınız parametre sayısı kadar "for" döngüsü açıp,
   AL/SAT noktalarını belirleyen stratejinizi bu for döngülerinin içine yazın
- Stratejinizi belirleyen kod parçasının hemen altına şu satırı ekleyin: Sistem.Optimizasyon("açıklama", x,y,z);
  - Değişken olarak sunup, taratarak en uygun değerini aradığınız parametreleri (x,y,z) for döngüleri içinde tanımlarken hangi iki seviye arasında taranmasını istiyorsanız belirtin
  - Son satırda, optimizasyon fonksiyonunun parantezi içinde, taratılan değişkenleri açtığınız for'ların sırasıyla ekleyin,
- Optimizasyon işlemi (hesaplaması) sürerken, diğer tüm işlemlerinizi, analizlerinizi yapmaya devam edin. Çünkü bu işlem, ayrı CPU çekirdeğinde yapılacak ve ana çalışma yapısını etkilemeyecektir.
- Optimizasyon işlemi sürerken, taranan parametrelere göre hesaplanan NETGETİRİ değerinin, taramada rastlanan/hesaplanan en büyük değeri ekranda gösterilir.

- Optimizasyon sürerken, taranan değerler için elde edilen getiri grafiksel olarak alt kısımda gösterilir.
- Optimizasyonu tüm kodun belli bir bar sayısı için veya iki tarih arasındaki barlar için yapılması tercih edilebilir.
- Opsiyonel olarak, sisteminizin MaxDD (Getirideki en büyük zarar miktarı ve tarihi) hesaplanmasını da isteyebilirsiniz. (Bu işlem, hesaplama süresini biraz daha fazla uzatır)
- Optimize edilmek istenen stratejiyi içeren sistemi istediğiniz herhangi bir grafik periyodu için hesaplatabilirsiniz.
- Sonuçları Excel sayfasına aktarabilirsiniz.
- İşlemi herhangi bir anda durdurabilir veya herhangi bir anda, o ana kadarki sonuçları işlemin bitmesini beklemeden göster diyebilirsiniz.
- FORMÜL butonuyla, optimizasyon formülünüzü açıp, değişiklikler yapıp kaydet diyerek yeniden yeni değerlere göre hesaplatma yapabilirsiniz.
- Optimizasyon işleminin, taranacak bar sayısı, kullanılan stratejinin içeriği ve karmaşıklığı, taratılan parametrelerin ilk ve son değerleri arasındaki fark, taratılıp en uygun değeri bulunmaya çalışılan parametre sayısı ve bilgisayarınızın işlemci ve Ram gücüne bağlı olarak zaman alacağını, bazı durumlarda saatler ve hatta günler sürebilecek kodları optimize etmeye çalışabileceğinizi unutmayınız.
- Sistemi yazdınız ama Optimizasyon kodunu yazamadığınız durumlarda <u>algo@idealdata.com.tr</u> adresine mail atarak yardım talebinde bulunabilirsiniz.
- Aşağıda birkaç değişik strateji için örnek optimizasyon formülü bulabilirsiniz.

### ÖRNEK-1: Kullanılan İndikatörler MA ve RSI

### Strateji: İki Ortalamanın kesişimi VE RSI'ın kendi ortalamasıyla kesişimi En uygun değeri aranan parametreler: MA1, MA2, RSI ve RSI2ın ortalamasının periyotları

```
var Kapanis = Sistem.GrafikFiyatSec("Kapanis");
var SonYon="";
for (int P1 = 3; P1 < 11; P1++)
{
   var MA1 = Sistem.MA(Kapanis , "Exp", P1);
   for (int P2 = 15; P2 < 20; P2++)
   ł
      var MA2 = Sistem.MA(Kapanis , "Exp", P2);
      for (int P3 = 6; P3 < 12; P3++)
      ł
         var RSI = Sistem.RSI(Kapanis , P3);
         for (int P4 = 2; P4 < 6; P4++)
         {
            var RSIAVR = Sistem.MA(RSI , "Exp", P4);
               for (int i = 1; i < Kapanis.Count; i++)</pre>
               Sistem.Yon[i] = "";
            // strateii
            for (int i = 1; i < Kapanis.Count; i++)</pre>
            ł
               if (RSI[i] > RSIAVR[i] && MA1[i] > MA2[i] && SonYon != "A") // alış
               {
                   Sistem.Yon[i] = "A";
                   SonYon="A";
               }
               else if (RSI[i] < RSIAVR[i] && MA1[i] < MA2[i] && SonYon != "S") // satış
                   Sistem.Yon[i] = "S";
                   SonYon="S";
               }
             }
```



} }

} } Sistem.Optimizasyon("MA1,MA2,RSI,AVR", P1 , P2 , P3 , P4);

istem Optimizasyo Başlangıç Bit Sembol Sistem Periyot 🗸 Açığa Satış 📝 MaxDD Hesapla Hesapla Formul 22.10.2014 01.11.2015 Excel VIP-X030 Optimizasyon\_MA\_RSI ▼ 5 Dk -🔽 Hızlı Mod En Büyük Kar En Büyük Zarar Toplam Zarar Toplam İşlem Ardışık Zarar No Açıklama Kar Zarar MaxDD Tarih Toplam Kar Kar Eden Zarar Eden Ardışık Kar MA1.MA2.RSLAVR, 7, 19, 6, 5 MA1.MA2.RSLAVR, 7, 19, 6, 4 MA1.MA2.RSLAVR, 7, 18, 10, 15 MA1.MA2.RSLAVR, 4, 18, 9, 3 MA1.MA2.RSLAVR, 4, 18, 6, 5 MA1.MA2.RSLAVR, 3, 18, 6, 4 MA1.MA2.RSLAVR, 3, 19, 11, 5 MA1.MA2.RSLAVR, 3, 19, 9, 2 MA1.MA2.RSLAVR, 3, 19, 9, 2 MA1.MA2.RSLAVR, 3, 19, 6, 2 1 16.750 04.06.2015 222.850 -200.40 1054 294 75 4.900 -7.100 -7.100 -6.750 -6.750 -6.750 -6.750 -6.750 -6.750 -6.750 -6.750 6.950 -7.900 04.06.2015 04.06.2015 04.06.2015 04.06.2015 04.06.2015 04.06.2015 04.06.2015 04.06.2015 04.06.2015 04.06.2015 222.850 222.275 245.200 245.100 245.025 250.175 249.750 249.650 249.125 2 3 4 5 6 7 8 9 10 21.65 17.500 12.500 -200.625 -225.050 1048 292 371 371 371 403 404 404 404 74 4.900 4.875 4.875 4.875 4.875 4.875 4.875 4.875 4.875 4.875 6.950 5.900 5.900 5.900 5.900 5.900 6.225 6.225 6.225 -7.900 -9.700 -9.700 -9.700 -9.700 -9.800 -9.800 -9.800 -9.800 20.150 19.750 19.700 19.550 18.675 18.625 18.325 18.325 1404 1015 1017 1019 1017 1129 1136 1136 1131 12.500 12.950 12.950 13.525 14.850 14.950 14.950 -225.050 -225.350 -225.475 -231.500 -231.125 -231.325 -230.850 1406 1408 1406 1570 1576 1576 1576 22.4500 10.0000 0.0000 -10.0000 -20.0000

### ÖRNEK-2: Kullanılan indikatör: Bollinger Strateji: Fiyatın, Bolinger alt ve üst bandını kesmesi Aranan değer: Bollinger indikatörünün parametreleri

```
var Veriler= Sistem.GrafikVerileri:
var Kapanislar= Sistem.GrafikFiyatOku(Veriler, "Kapanis");
var SonYon = "";
for (int x = 5; x < 15; x++)
{
   for (double y = 0.1; y < 3; y + = 0.1)
   {
      var BolingerAlt = Sistem.BollingerDown(Veriler, "Exp", x, y);
      var BolingerUst = Sistem.BollingerUp(Veriler, "Exp", x, y);
      for (int i = 1; i < Veriler.Count; i++)</pre>
      Sistem.Yon[i] = "";
         for (int i = 1; i < Sistem.BarSayisi; i++)</pre>
         {
             if (Kapanislar[i] > BolingerUst[i])
             {
                if (SonYon != "A")
                {
                   Sistem.Yon[i] = "A"; // alış
                   SonYon = "A";
                }
             if (Kapanislar[i] < BolingerAlt[i])</pre>
             {
                if (SonYon != "S")
                {
                   Sistem.Yon[i] = "S"; //
                   SonYon = "S";
                }
            }
         }
      Sistem.Optimizasyon("BollingerParams",x,y);
   }
}
```

## ideal

Sisten	n Optimizasyon														-   P   X
K/2	z -5.13	MAX	50.23						STOP	Göster	Temiz	le			
BollingerParams , 11 , 1 Optimizasyon_Bolinger , 5 Dk															
No		Açıklama		Kar Zarar	MaxDD	Tarih	Toplam Kar	Toplam Zarar	Toplam İşlem	Kar Eden	Zarar Eden	En Büyük Kar	En Büyük Zarar	Ardışık Kar	Ardışık Zarar
1 2 3 4 5 6 7	BollingerParam BollingerParam BollingerParam BollingerParam BollingerParam BollingerParam BollingerParam	s, 6, 1.8 s, 6, 1.7 s, 5, 1.7 s, 5, 1.6 s, 5, 1.1 s, 5, 0.9 s, 5, 0.1		50.225 38.225 32.150 16.925 12.975 9.275 -0.175	12.625 11.550 11.000 13.000 15.450 15.325 15.350	30.09.2015 13.07.2015 24.07.2015 13.07.2015 14.05.2015 27.04.2015 05.06.2015	134.625 145.800 131.650 143.175 232.475 254.750 337.975	-84.400 -107.575 -99.500 -126.250 -219.500 -245.475 -338.150	256 396 283 415 1396 1848 4174	117 164 115 160 488 602 1184	137 230 162 248 872 1181 2792	5.550 5.550 6.300 6.125 3.150 3.350 2.250	-3.825 -2.625 -3.775 -3.175 -6.750 -6.500 -6.500	10.975 7.550 9.425 6.500 4.725 5.050 3.900	-6.875 -4.825 -5.725 -6.950 -10.000 -9.750 -7.125
															50.2250
															0.0000
															0.0000
															-50.0000

**PERFORMANS:** Sistem ve Robot kullanıcılarının neredeyse başucu ekranlarından biri de Performans Analiz aracıdır.

Yazdığınız, optimize ettiğiniz, güncellediğiniz, sık sık hareketlerini gözlemlediğiniz sisteminize ait bütün analiz raporu detaylarına bu tablodan ulaşabilirsiniz.

- Sisteminizi, sistemin uygulanmasını istediğiniz sembolünüzü ve baz alınacak grafik periyodunu seçip, bu sistem geçmişte neler yapmış sorusunun cevabını pek çok detay ile birlikte görebilirsiniz.
- İsterseniz, sistemi uygulama başladığınızdaki bakiye değerini girip, sistem uygulansaydı son bakiyeniz ne olurdu bilgisi görün. (bakiyeye sıfır girip, sadece puan/tl bazında kayıp kazanç da görebilirsiniz.)
- Açığa satış olmayan (örneğin AL/KAPAT, AL/KAPAT) şeklinde işleyen bir HİSSE SENEDİ için analiz yapıyorsanız, AÇIĞA SATIŞ kutusundaki işareti kaldırın
- Her bir sinyalde kaç lot işlem yapılacağı bilgisini de kullanıp, portföyün sanal simülasyonunu daha net yapın
- Sonuçları Excel sayfasına aktarabilirsiniz.
- Sistemi belli bir bar sayısı için taratabilirsiniz.
- Sisteminizin, gireceğiniz iki tarih arası için, MaxDD (en büyük zararı yaşadığı tarih ve değer) sorgusunu da yapabilirsiniz.
- Performans tablosunun ana gövdesinde, geçmişten günümüze (yada formülde hesaplattırılan tarihler arasında) yaptığı tüm işlemlerin detaylarını (işlem yönü, tarihi, işleme girdiği fiyat, işlemin kazancı veya zararı, o işlem sonrası bakiyenin son durumu) gözlemleyebilirsiniz.

Sistem Performansı											-		×
Sembol	Sistem		Perivot		İlk Ba	kive Lot			Başlangıç	Bitiş			
VIP-X030	Sistem1		• 60 Dk	•	100	1		🔽 Açığa Satış	25.05.1990	04.06.2020	Yenile	Exc	el
Baslangic Bakiyesi	100.000	No	Yön	In	nt	Acılıs Tari	hi	Acilis Evit	Kananıs Tarihi	Kananis Evt	Kar / Zarar	Bakiye	
Son Bakiye	176.350	1	Abe		~ 1	22.07.2011	17:00	73 150	00.08.2011.00:00	64 375	8 775	01 225	
Toplam Getiri	76.350	2	Satis		1	09.08.2011.	10.00	64 375	07.09.2011.11:00	67,650	3 275	87.950	
% Getiri	76.35	3	Alix		1	07.09.2011 1	11:00	67,650	20 10 2011 13:00	69,700	2.050	90,000	
		4	Satis		1	20 10 2011 1	13:00	69,700	05 12 2011 13:00	67 325	2.050	92 375	=
Toplam İşlem Sayısı	127	5	Abz		1	05 10 2011 :	13:00	67,325	12 12 2011 15:00	64 100	3 3 25	89.150	
Karlı İşlem Sayısı %	34.00	6	Satis		1	12 12 2011 :	15:00	64,100	16 01 2012 11:00	63.425	- 5.225	80.875	
Kazandıran İşlem Sayısı	44	7	Alix		1	16 01 2012 :	11:00	63,425	06.03.2012.11:00	71 375	7 950	03.025	
Kaybettiren İşlem Sayısı	83	8	Satis		1	06.03.2012 :	11:00	71 375	14 03 2012 17:00	73,400	-2.025	95 750	
Kazandıran Toplam	310.300	9	Alis		1	14.03.2012 :	17:00	73,400	09.04.2012.17:00	73.600	0.200	95.950	
Kaybettiren Toplam	-233.950	10	Satic		1	09.04.2012	17:00	73,600	08.06.2012.17:00	69.025	4 575	100 525	
Net Kar	76.350	11	Alix		1	08.06.2012 :	17:00	69.025	24.07.2012.10:00	74 375	5 350	105.875	
Profit Factor	1.33	12	Satis		1	24.07.2012	10.00	74 375	26.07.2012.16:00	77,500	-3 125	102.750	
En Büyük Kar	30.925	13	Aliz		1	24.07.2012	16:00	77,500	28.09.2012 13:00	91.950	- J. 12 J	107,700	
En Büyük Zarar	-8.775	14	Satic		1	28.09.2012	13:00	81.950	08 10 2012 13:00	82.975	1.025	106 175	
Ardısık Kar Savısı	4	10	Alic		1	09 10 2012 :	12:00	82.075	20.11.2012.16:00	02.375	-1.025	111 550	
Ardısık Kar Miktarı	34.175	16	Satur		1	20.11.2012	16:00	99.250	20.11.2012 10:00	00.000	1.650	100.000	
Ardısık Zarar Savısı	13	17	Aliz		1	20.11.2012	11:00	90,000	20.01.2012.17:00	09.450	-1.050	119 350	
Ardısık Zarar Miktarı	-48.450	10	Cotty		1	20.01.2012 :	17.00	09.450	04.02.2012 16:00	00,705	1.75	117.075	
Alım İslemi Savısı	64	10	alic		1	04.02.2012 :	16.00	00,735	00.04.2012.14:00	101 500	-1.275	110.000	
Alım İslemi Yüzdesi	50.39	20	Satur		1	09.04.2013	14:00	101 500	12 04 2012 14:00	101.500	2,100	115 750	
Satım İslemi Savısı	63	20	Alic		1	12.04.2012	10.00	101.500	19.04.2013 15:00	104.000	-5.100	112,750	
Satım İşlemi Yüzdesi	49.61	21	Catur		1	10.04.2013	15:00	104.000	24.04.2013 15:00	104.300	- 3,000	109.950	
		22	ally a		1	24.04.2012 :	15.00	104,000	24.04.2013 15.00	107.405	-3,300	111.075	
May DD F	Hesanla	23	Satur		1	24.04.2013	16:00	104.300	17.07.2013 16:00	04.425	12 000	124.075	
	resupra	24	Alic		1	17.07.2012	16:00	94.425	25.07.2012.11:00	99,700	6 135	119.950	
26.12.2018	26.12.2029	20	Catur		1	25.07.2013	11:00	94,425	13 09 2013 16:00	00.300	-0.125	115 500	
		20	Abz		1	13.08.2013	16:00	91.650	16 08 2013 17:00	91.050	-5,550	113 000	
		27	Satis		1	16.08.2012 :	17:00	90.050	11.00.2013 1/:00	30.000	-1.000	115 275	
		20	al			11.00.2013	14.00	90,050	05 11 2012 14:00	00.075	2.375	110.270	

**SEMBOL BAZINDA SİSTEM KIYASLA:** IDEAL üzerinde yazdığınız sistemleri birbirleriyle en hızlı kıyaslama aracı SEMBOL BAZINDA SİSTEM KIYASLA analiz aracıdır. Sistem menüsünden SEMBOL BAZINDA SİSTEM KIYASLA satırına basarak açabileceğiniz bu analiz ekranında, istediğiniz kadar sistem seçerek, belirlediğiniz sembolün belirlediğiniz periyodu için sistemlerin performanslarını hesaplatır ve aşağıdaki analiz kalemleri bazında sonuçları görürsünüz

- KAÇ ADET İŞLEM (SİNYAL) ÜRETMİŞ,
- İŞLEMLERİN KAÇI KAZANDIRAN YÖNDE İŞLEM OLMUŞ,
- İŞLEMLERİN KAÇI KAYBETTİREN YÖNDE İŞLEM OLMUŞ,
- HER BİR SİSTEM 5 BAZINDA NE KADAR KAZANDIRMIŞ/KAYBETTİRMİŞ,
- HER BİR SİSTEM PUAN BAZINDA NE KADAR KAZANDIRMIŞ/KAYBETTİRMİŞ.

Sembol Bazında Sistem Kıyasla ekranında, sembolü, grafik periyodunu veya bar sayısını değiştirerek hızlıca yeni tercihler için sonuçları görebilirsiniz.

Sol bölgeden yeni bir sistem ekleyebilir veya var olan bir sistemi çıkarabilirsiniz. Sistem ekleme/çıkarma yaptıktan sonra yeniden HESAPLA butonuna basmanız yeterlidir.

Kıyaslama sonuçlarını Excel sayfasına aktarabilirsiniz.

HİÇBİRİ butonuna basarak, seçili sistemlerdeki seçim işaretini kaldırabilir, TÜM butonuna basarak arşivdeki tüm sistemleri seçtirebilirsiniz.

Sistem Kıyasla			100				X
Tüm Hiçbiri		Sembol Per VIP-X030 60	riyot Bar Sayısı Dk - 1000	Hesapla	Excel		
Robot_Fiyat_Hisse_TekYon2	No	Sistem	İşlem	Kazandıran	Kaybettiren	% Getiri	Puan
📃 Robot_Fiyat_Vadeli_TekYon	1	_TEST5	59	33	26	26.29	23.200
🔲 Robot_Flat	2	A1_SISTEM	49	19	30	17.45	16.375
🔲 Robot_Flat2	3	A2_SISTEM	26	9	17	-9.06	-8.775
Robot_FX_02	4	BolingerSys	94	41	53	30.12	25.800
Robot_FX_02_rOBOT	5	HHV_LLV_Sistem	12	5	7	-1.32	-0.300
Robot_H_Test	6	MA	12	3	9	-7.78	-7.550
🔲 Robot_Halkb_Garan	7	TOMA	16	6	10	-7.39	-6.850
🔲 Robot_HerDakikaBirEmir	8	Walker	17	4	13	-19.91	-20.450
Robot_HerDakikaBirEmir_SEZ1							
Robot_HerDakikaBirEmir_SEZ2							
🔲 Robot_HerDakikaBirEmir5							
🔲 Robot_HerDakikaBirMesajsifreli2							
Robot_Hisse_3							
Robot_Hisse_4							
Robot_HL							
Patrat United to							

**SON POZİSYONLAR:** IDEAL üzerinde belirlediğiniz stratejiye dayalı AL/SAT sinyalleri üreten bir sistem yazdınız diyelim.

Bu sistemin kullandığı stratejiye göre tüm hisselerde/sembollerde SON DURUM/YÖN nedir görmek için SON POZİSYONLAR analiz aracı kullanılır.

Örneğin, 5 ve 50 periyotlu iki hareketli ortalama kesişince AL/SAT üreten MA sistemini, belirlediğiniz bir senet/sembol grubu için taratıp, bu sisteme göre bu gruptaki sembollerin son durumu/yönleri nedir, ne zaman hangi fiyattan sinyal üretilmiş, son fiyat ve bu fiyata göre son yöndeki kar/zarar nedir bir tablo olarak görebilirsiniz.

SON POZİSYONLAR analiz aracı, oldukça yararlı ve pratik bir analiz aracıdır ve şu imkânları sunar;

- Seçtiğiniz sistemi istediğiniz herhangi bir grafik periyoduna göre hesaplattırabilirsiniz.
- Sol üst köşedeki kutudan, istediğiniz herhangi bir Sistemi seçebilirsiniz
- Tabloya aktarılan sonuçları Excel'e aktarabilirsiniz.
- Seçtiğiniz Sistemin EN SON NE ZAMAN SİNYAL ÜRETTİĞİ filtresi yapabilirsiniz. (Örn Son 60 dk içinde bir sinyal üretilmiş senetleri görmek için 60 DK seçeneği işaretlenir. Sadece Bugün sinyal üretilmişleri görmek için BUGÜN seçeneği)
- Sistemin tarayacağı sembol grubunu değiştirebilirsiniz. Üstteki XU-100 butonuna basarak hazır bir grup seçebilir veya kendi oluşturduğunuz bir sembol listesi içindeki kodları taratabilirsiniz.(kendi özel sembol listelerinizi nasıl yaratacağınızı bilmiyorsanız teknik servisimizden yardım isteyebilirsiniz.)

MA		▼ 5 Dk	▼ XU-100		Excel		
© 60 [	Dk 🔘 Bugün 🔘 B	ir Hafta 🛛 🔊	Bir Ay 🔘 Tüm				
No	Sembol	Yön	Tarih	Fiyat	Son Fiyat	Fark	%
1	ZOREN	Alış	22.10.2015 09:35	1.62	1.60	-0.02	-1.23
2	TKNSA	Alış	22.10.2015 11:05	6.56	6.55	-0.01	-0.15
3	NETAS	Satiş	22.10.2015 11:00	11.20	11.20	0.00	0.00
4	METRO	Satiş	22.10.2015 10:35	0.68	0.68	0.00	0.00
5	KOZAL	Alış	22.10.2015 11:00	20.05	20.00	-0.05	-0.25
6	KARTN	Alış	22.10.2015 09:55	238.00	237.50	-0.50	-0.21
7	ISCTR	Satiş	22.10.2015 10:25	5.03	5.03	0.00	0.00
8	IPEKE	Alış	22.10.2015 11:00	1.93	1.93	0.00	0.00
9	HLGYO	Satiş	22.10.2015 10:50	1.02	1.03	-0.01	-0.98
10	GARAN	Alış	22.10.2015 09:40	7.62	7.61	-0.01	-0.13
11	FROTO	Satiş	22.10.2015 10:40	33.80	33.90	-0.10	-0.30
12	ECZYT	Alış	22.10.2015 09:45	10.14	10.09	-0.05	-0.49
13	ECILC	Satiş	22.10.2015 10:30	2.54	2.54	0.00	0.00
14	DOHOL	Alış	22.10.2015 09:35	0.60	0.60	0.00	0.00
15	DEVA	Alış	22.10.2015 09:35	3.73	3.72	-0.01	-0.27
16	BIMAS	Alış	22.10.2015 10:05	58.65	58.65	0.00	0.00
17	BAGES	Alış	22.10.2015 10:15	14.05	14.00	-0.05	-0.36
18	ASELS	Satiş	22.10.2015 11:00	14.65	14.70	-0.05	-0.34
19	ANACM	Satiş	22.10.2015 10:40	2.28	2.29	-0.01	-0.44
20	ALGYO	Alış	22.10.2015 10:50	24.90	24.90	0.00	0.00
21	AKBNK	Alis	22.10.2015 09:40	7.55	7.50	-0.05	-0.66

**MULTİ PERFORMANS EĞRİSİ:** IDEAL üzerinde birden fazla sistem yazdınız ve bu sistemlerin hangisi daha iyi, hangisi daha karlı görmek istiyorsunuz. Bunu tespit etmenin çokça yolu var. En doyurucu analizi yapabileceğiniz özellik, sistemlerin getiri eğrilerini tek bir grafik üzerinde karşılaştırmalı olarak görmenizi sağlayan Multi Performans Eğrisidir. Sistem menüsünden MULTİ PERFORMANS EĞRİSİ satırına basarak açabileceğiniz bu analiz ekranında, istediğiniz kadar sistem seçerek birbiriyle kıyaslayabilir ve getiri eğrilerini bir grafik üzerinde gözlemleyebilirsiniz.

- Ekle butonuna basarak, kıyaslama listesine sistem ekleyebilirsiniz.
- Çıkart butonuna basarak bir sistemi listeden kaldırabilirsiniz.
- Her sistemin getiri eğrisi için renk seçebilirsiniz.
- Bir sistemi o an için kıyaslama ekranında görmek istemiyorsanız (listeden tamamen çıkarmadan) sadece başındaki kutudaki işareti kaldırabilirsiniz.
- Seçili tüm sistemlerin getiri eğrilerini istediğiniz herhangi bir sembol (senet, endeks, parite, vadeli kod vs.) bazında bakabilirsiniz. Orta üst kısımdan bir sembol kodu girmeniz yeterli
- Seçili Sistemleri, seçili sembolün istediğiniz herhangi bir periyotlu grafikleri bazında kıyaslayabilirsiniz. (orta üst kısımdaki periyot seçim kutundan)
- Tüm sistemlerin getirileri orta üst kısımda yazılı bar sayısı kadar performans sonucu döndürür. Son 5000 bar için sonuç görmek isterseniz bar sayısı kutusuna 5000 yazıp ENTER'layın.
- Sembol, periyot veya bar sayısı değiştirip HESAPLA butonuna basmak yeterlidir.

- Panelin sağ kısmında seçilen sembolün seçilen periyotlu grafiği, onun alt kısmında ise seçili sistemlerin GETİRİ grafikleri gösterilir.
- Sol bölgede hangi sistem adına tıklanırsa, sağ taraftaki grafik üzerinde, o sistemin AL/SAT noktaları gösterilir.



### iDeal Sistem Modülünü kullanırken bilinmesinde yarar olan Temel C# Yazılım dili kuralları

iDeal Sistem modülünde C# yazılım dili kurallarına göre formül yazılır. iDeal tarafından sunulan hazır fonksiyonlarla birlikte, C# dilinde yazılım geliştirirken kullanılabilecek hemen hemen tüm C# komutlarını da kodlarınızda kullanabilirsiniz.

Yazılımcılık konusunda deneyimli olan kullanıcılar C# kodlamanın elverdiği hemen hemen tüm imkânları kullanarak formül yazabilirler. Örneğin bir sayının karesi C# dilinde Math.Pow(x,2) şeklinde alınır ve aynı yazım şekli ideal formüllerinde de kullanılabilir.

IDEAL Sistem Modülünde bir sistem/formül yazma aşamalarında ihtiyaç duyabileceğiniz temel bilgileri ve önemli ipuçlarını aşağıda sıraladık.

### IDEAL Üzerinde Formül Yazmanın Genel Akış Diyagramı

- Kullanıcı tarafından girilecek tüm önceden tanımlı bilgiler (sayılar, fiyatlar, periyotlar, semboller, süreler vs) birer değişken olarak tanımlanır. (değişkenler ve tanımlanmaları aşağıda açıklanacaktır)
- IDEAL üzerinden elde edilecek tüm veriler ve bilgiler (yüzeysel, derinlik, grafik veya indikatör verileri) ilgili fonksiyonlarla okutulup (Her fonksiyonun kullanım şekli ve örnekleri IDEAL SİSTEM FONKSİYONLARI menüsü altında anlatılmıştır.) birer değişkene atanır.

SEZAİ KILIÇ

\_

- Formül süresince bazı hesaplamalar sonucu elde edilecek yeni bilgiler veya listeler için de birer değişken tanımlanır.
- Her değişken tanımlandığında, değişkene bir değer atamak şarttır. (bu değer BOŞLUK (NULL) veya SIFIR olabilir.
- Değişken tanımlamak için değişkenin tipini yazıp bir boşluk bırakıp, tanımlanan değişkene bir isim verilir. (değişken isimleri Türkçe karakter veya noktalama işaretleri içeremez, birden fazla kelimeden oluşamaz ve büyük küçük harfe duyarlıdır.)
- Değişken tanımı bir atama ile (değişkeni bir değere eşitlemek ile) tamamlanır. Bu durumda değişken tanımlanan satır

### DEGISKENTİPİ (BOŞLUK)DEGISKENE VERİLEN İSİM = DEĞER

- Son olarak satırın tamamlandığı bilgisi için satır sonuna NOKTALI VİRGÜL konur. (bazı özel durumlar hariç her satırın tamamlandığı bilgisi için satır Noktalı Virgül ile bitirilir.)
- Tam satır şu şekilde oluşur:

### DEGISKENTİPİ (BOŞLUK)DEGISKENE VERİLEN İSİM = DEĞER;

### Değişken Tipleri:

Sadece C# değil, diğer programlama dillerinde de en sık kullanılan elemanlar değişkenlerdir. Bilinen/bilinmeyen tüm VERİ değişkenlere atanır ve tüm hesaplamalar/kıyaslamalar ve yeni veri elde etmeler Değişkenler üzerinden elde edilir. Böylece kodlama anlaşılır ve sadeleşir. En sık ihtiyaç duyulan değişken tipleri aşağıda gösterilmiştir:

- **var** = Global/Genel değişken tipi. Bu türde değişkenlere bir değer verildiği zaman, o değerin (sayı, tarih, metin vs) formatına uygun bir tipte veri saklar.
- int = Tam Sayı cinsinde veriler için kullanılır (1,3,10,2500 vs)
- float = kesirli sayısal değerler için kullanılır
- single = çok sayıda ondalık basamak içeren reel sayılar için kullanılır
- **double** = çok sayıda ondalık basamak içeren reel sayılar için kullanılır
- string = yazı/metin formatındaki bilgiler için kullanılır

### ÖRNEKLER:

var x = 0; var y = ""; int adet = 25; float fiyat = 1.52f; double oran = 0.1254; string mesaj = "merhaba ideal";

### Eğer ise / Değil ise ve Mantıksal Operatörler (if / else if):

Sistem, Robot, indikatör formülü yazarken en çok ihtiyaç duyulan işlerden biri de bazı koşulların olup olmadığını kontrol etmektir. EĞER şöyle İSE böyle yap gibi kontrolleri **if** (EĞER) ve **else** (DEĞİLSE) komutlarıyla yaparız.

Bu tür kontrolleri yaparken de matematikteki mantıksal operasyonları kullanılırız. İhtiyaç duyacağınız mantıksal operatörler ve kullanım şekilleri aşağıda gösterilmiştir.

- x > x x BÜYÜKTÜR y
- x < y x KÜÇÜKTÜR y
- x == y x EŞİTTİR y
- x != y x EŞİT DEĞİLDİR y
- x >= y x BÜYÜKTÜR VEYA EŞİTTİR y
  - x <= y x KÜÇÜKTÜR VEYA EŞİTTİR y
- x && x XEy
- x || x x VEYA y (Bu işaret ALTGR ile TİRE/EKSİ işaretine basılarak yapılır)

Bu durumda bir koşul sağlanıyorsa bir işi, sağlanmıyorsa başka bir işi yaptırtmak için aşağıdaki gibi bir kod satırı gerekir; (x değeri y değerinden **küçükse**, "a" değerini eksi 5, **değilse** 5 yap)

```
if (x < y)
a = -5;
else
a = 5;
```

\_

Yazılım dillerinde (aksini yapacak komutlar kullanılmamışsa) kodlar satır satır okunur ve işletilir. Yukarıdaki örneği çalıştıran uygulama önce "if" satırına gelecek ve if parantezi içindeki koşula bakacak.

Koşul içindeki ifade DOĞRU ise ALT SATIRA geçecek ve orada söylenen işi yapacak, "else" satırını ve o satır altına yazılan işi direk atlayacaktır. Tam tersi durumda, if komutunun içindeki koşul sağlanmıyorsa, uygulama bu kez else (DEĞİLSE) satırında tanımlanan işi yapacaktır.

Eğer koşul sayısı birden fazla ise (Koşul1 sağlanıyorsa bir iş, koşul2 sağlanıyorsa bir başka iş, koşul3 sağlanıyorsa bir başka iş...... Hiçbiri sağlanmıyorsa bir başka iş) şeklinde seri olarak koşullar varsa komutun kullanım şekli şöyle olur:

```
EĞER (Koşul1)
İŞ 1;
YOK EĞER (Koşul2)
İŞ 2;
YOK EĞER (Koşul3)
İŞ 3;
HİÇBİRİ
İŞ 4;
```

Bu algoritmanın C# dilince yazımı şöyledir;

Eğer, IF/ELSE koşullarının altında yaptırılacak işlemlere ait satır sayısı BİRDEN FAZLA İSE, tüm bu satırlar iki tane parantez (güzel parantez) içine alınmak zorundadır.



### Listeler Tanımlamak:

IDEAL üzerinde formül yazarken genellikle LİSTELER kullanılacaktır. LİSTE birden fazla elemanı olan bir tablo gibi düşünülebilir.

Zira Sistem/Robot formülleri genellikle hisse senedi, parite, endeks veya vadeli sözleşmelerin tarihsel GRAFİK VERİLERİ üzerinde kurulan strateji ve hesaplamalar içerir.

GRAFİK VERİSİ, belirli bir zaman (periyot) boyunca her bir zaman dilimine karşılık bir veya birden fazla veri içeren (BARLAR) bir LİSTEDİR.

Aynı şekilde İndikatörler de birer listedir ve her bir BARA karşılık bir değer içerirler. Zaman ekseninde (grafikteki yatay eksen) her bir zaman dilimine karşılık bir değeri olan listeler aslında aynı zamanda birer ÇİZGİDİR.

Bir hissenin AÇILIŞ, KAPANIŞ, YÜKSEK, DÜŞÜK fiyatları, Bir indikatörün değerleri, bir yatay çizgi vs. Hepsi birer listedir.

IDEAL Sistem modülündeki pek çok fonksiyon, bir liste formatında veri okur ve kullanıcıya sunar.

Kullanıcılar, daha sonra hesaplanacak verileri atayacakları listeleri kodlarının başında, bir isim vererek sisteme tanımlarlar. Liste tanımlarken global değişken tipi olan "var" kullanılır.

Diğer (sayı veya metin tipindeki) değişkenlerde olduğu gibi, listeleri tanımlarken de eleman listenin içindeki her bir elemanın başlangıçtaki değeri belirtilir.

Örnek: var Listem1 = Sistem.Liste(x);

Bu örnekte Listem1 isimli bir liste tanımlanır ve listenin tüm elemanları başlangıçta x değeridir. (Boş listede x = 0)

Örneğin grafiğe belli bir fiyat seviyesinde bir yatay çizgi çizilecekse ve o çizgiye SEVIYE adını verelim)

#### var SEVIYE = Sistem.Liste(95.400); Sistem.Cizgiler[0].Deger = SEVIYE;

Satırlarını yazmak yeterli olacaktır.

(Kütüphane Ana sayfamızdaki IDEAL Sistem Fonksiyonları kısmından, Liste ve Çizgiler isimli fonksiyonları kullanım detayları incelenebilir.)

### Listeler Üzerinde işlem (hesaplama) Yapmak:

## ΠΕΔL

Listeler üzerinde işlem yapmak için genel yazılım kuralı olarak birer isim verip değişkenler tanımlar ve işlemleri bu değişkenler üzerinden yaparız. Fakat Metastock ve benzeri programlarda alışılagelen yöntemle, doğrudan listelerin kendileri işleme tabi tutulamaz. Listeler, birden fazla eleman içerirler ve her türlü işlem (matematiksel işlem, mantıksal kıyas vb.) LİSTELERİN ELEMANLARI arasında yapılabilir.

Bir listenin her hangi bir elemanı, o listenin bir INDEX'i ile belirtilir. Yani: Listenin x numaralı elemanı, liste adının yanına köşeli parantez içinde x yazılarak belirtilir.

ÖRNEK: Bir grafikte 1350 numaralı (ilk bar en eski bar) barın kapanış değeri **KAPANISLAR[1350]** olarak belirtilir ve bu şekilde kullanılır.

ÖRNEK2: KAPANISLAR isimli listeye, bir grafiğin barlarının kapanış fiyatları alınsın. 350 ve 500 numaralı barların kapanış fiyatlarının toplamı X olsun. X değerini bulalım;

```
var KAPANISLAR = Sistem.GrafikFiyatSec("Kapanis");
var X = KAPANISLAR[350] + KAPANISLAR[500];
```

```
ÖRNEK3: 100 numaralı bar kapanışı, önceki bar kapanışından büyükse, x değerini 1 arttır
var KAPANISLAR = Sistem.GrafikFiyatSec("Kapanis");
var x = 0;
```

```
if (KAPANISLAR[100] > KAPANISLAR[100-1])
    x = x+1;
```

(Burada 99 yazmak yerine özellikle 100-1 yazdık köşeli parantez içine. Aslında aynı şey ama sonraki satırlarda anlatılacak diğer örnek ve açıklamalara yatkınlık olsun diye 99 yerine 100-1 yazdık.)

### Döngüler:

IDEAL de formül yazarken en çok kullanılacak işlemlerin başında DÖNGÜLER gelir. DÖNGÜ, bir işlemi/hesaplamayı birden fazla kere yaptırtmak için kullanılır.

Yani örnek olarak, MA5 ile MA100 değerlerini kıyaslayacaksak bu kıyaslamayı HER BAR BARDAKİ DEĞERLERİ için yapmak gerekir. 5'lik ve 100'lük ortalamaların birbirinden büyük/küçük olduğu ya da birbirini kestiği noktalar

Birer BAR'dır ve bu kontrol HER BAR için yapılınca, sinyalin/koşulun hangi Bar'da olduğu, o barın *fiyat/zaman/barno* gibi bilgileri de elde edilmiş olur.

Yine örneğin, bir sembolün grafiğindeki Yüksek ve Düşük değerlerinin toplamının yarısına ihtiyacımız varsa, Yapılacak olan matematiksel işlem olan (H + L) / 2 işlemi HER BAR için yaptırılır ve HER BAR için çıkan sonuç, bir listenin o bara karşılık gelen elemanına yazılır. Bu işlemler DÖNGÜ içerisinde yapılır.

KALIP olarak her zaman kullanabileceğiniz döngü formatı şöyledir;

```
for (int x = 1; x < 100; x++)
{
//DÖNGÜ BOYUNCA YAPILACAK TÜM İŞLEMLER
}
```

### Bu kalıpta "for" kelimesi yazılıp bir parantez açılır ve 3 adet bilgi parantezin içine sırayla girilir ve parantez kapatılır

Parantezin içindeki ilk bilgi (x), döngünün uygulanacağı listenin başlangıç değeri (veya kaç numaralı elemanından itibaren). Yazının en başında bahsettiğimiz bazı C# kuralları burada tekrar devreye girer:

- Her bilgi satırı NOKTALI VİRGÜL ile biter
- Döngünün başlangıç ve bitiş değerlerini bir değişken kontrol eder. Bu nedenle değişken tanımlama kuralı gereği DEGISKENTIPI (BOŞLUK) DEGISKENADI = DEĞER; kalıbı ile ilk döngü bilgisi verilmiş olur. (Bar numaraları birer birer artacağından, bize TAMSAYI değişkeni yeterli bu yüzden x değişkenimizin tipini "int" yaptık)
- Parantezin içine girilmesi gereken ikinci bilgi, döngünün ne zaman sona ereceğidir. Yani örneğimizdeki "x" kaç olana kadar bu işlem yapılsın. (Örnekte x = 99 olana kadar işlemler sürer. 100'ün de dahil olması için KÜÇÜK EŞİT ifadesi yazılmalı)
  - Bu da bir bilgi alanıydı ve diğer bilgi alanları gibi **NOKTALI VİRGÜL İLE bitirildi.**
- Parantezin içindeki son bilgi, ilk değerden son değere kadar dönecek olan işlemler boyunca, x değeri kaçar kaçar artacak. Birer birer artacaksa değişken adının yanına iki tane "+" işareti konur (x++ ile x=x+1 aynı şeydir.)
  - Döngüyü bar bar yapmayıp, başka bir değere göre yapıyorsak ve tarama birer birer artan değerler için değilse, döngüdeki x değişkenini integer (int) olarak değil double (reel sayı) olarak tanımlarız.
  - Örneğin for (double x=0.25; x <= 10; x+=0.05) Bu döngüde, işlemler, x değerinin 0.25'ten başlayıp 10 olana kadar (10 dahil) ilerlenir ve her seferinde x değeri 0.05 arttırılır.</li>
- for ve if/else komut satırlarında (ve bu bloklar için açılmış olan güzel parantezlerin olduğu satırlara) NOKTALI VİRGÜL KONULMAZ.

Artık döngü kalıbının içinde, listeler arasında kıyaslama, hesaplama ve bir başka listeye değer atama işlemleri yapılabilir.

Burada dikkat edilecek tek nokta, yukarılarda da birkaç kez vurguladığımız gibi, işlem, kıyas veya atamayı yaparken listenin hangi elemanı olduğunu belirterek (idexini de yazarak) yapmaktır. Yani LISTEADI[x] şeklinde kullanmak.

ÖRNEK-1: Bir grafiğin High, Low ve Close değerlerini toplayıp 3'e bölelim ve sonucu ekranda çizgi olarak görelim:

(Her satırın yanına, o satırda ne yapıldığına dair notlar düşülmüştür.)

var V = Sistem.GrafikVerileri ; //Kodun uygulanacağı kodun, o an seçili periyodu için tüm grafiği (oku ve V isimli listeye al)

var C = Sistem.GrafikFiyatOku(V, "Kapanis" ); //V listesinden (OHLC barlardan) KAPANIŞ FİYATI listesini oku ve C isimli listeye al

var H = Sistem.GrafikFiyatOku(V, "Yuksek" ); // V listesinden (OHLC barlardan) HIGH FİYATI listesini oku ve H isimli listeye al

var L = Sistem.GrafikFiyatOku(V, "Dusuk" ); // V listesinden (OHLC barlardan) LOW FİYATI listesini oku ve L isimli listeye al

var Sonuc = Sistem.Liste(0); //Boş (bütün elemanları sıfır olarak tanımlanan) bir liste tanımla ve listeye "Sonuc" adını ver



for (int i = 1; i < V.Count; i++) //1 numaralı bardan başla, V listesinin eleman sayısına kadar , "i" değerini BİRER BİRER arttır

Sonuc[i] = ( H[i] + L[i] + C[i] ) / 3; //her bir adımda (her bir "i" değeri için) H,L ve C listelerinin o bara denk gelen elemanlarını H[İ], L[i] ve C[i] topla ve 3'e böl. Bunu boş listemizin "i" numaralı satırına yaz (Sonuc[i] ) }

//DÖNGÜ bitince her bar için hesaplanıp Sonuc listesine yazma işi tamamlanmış olur ve Sonuc isimli liste bu satırda artık hesaplardan elde edilen değerlerle doldurulmuştur.

Sistem.Cizgiler[0].Deger = Sonuc; //Sonuc isimli listeyi grafikte bir çizgi olarak göster. (yazılan formüle LISTE\_DONGU ismi verilip, çift tıklanarak grafiğe uygulanmıştır.)



NOT: Grafik barlarını tarayarak yapılan işlemlerde, yukardaki kalıp, 1'den başlar ve bar sayısı kadar ileriye doğru ilerler. Yani ilk hesaplama/atama işlemi EN ESKİ TARİHLİ bar için yapılır.

Eğer işlemi ters sırayla yapmak istersek (önce en son bar, sonra birer birer geriye doğru en eski bara gitmek) Kalıp şöyle olur: for (int x = 100; x > 1; x--)

### **IDEAL SISTEM KÜTÜPHANESI VE SISTEM FONKSIYONLARI**

Kılavuzumuzun buradan sonraki kısmında, formüller yazarken kullanabileceğiniz ve tamamı Sistem.XXX şeklinde yazılan bütün sistem elemanları tek tek anlatılacaktır. Bu sistem elemanlarının bazıları verilere erişim, bazıları çeşitli aksiyonlar almak (emir gönder, sorgu ekle, mesaj çıkar vs.) bazıları ise iDeal içindeki indikatörleri çağırmak için kullanılır. LİSTE ve VERİLER ile indikatör Komutları Kullanımı:

İndikatörlerin hemen hemen tamamında iki üç kullanım yöntemi vardır. Örneğin RSI için aşağıdaki gibi üç çeşit fonksiyon kullanılabilir.

- Sistem.RSI(14);

- Sistem.RSI(Liste,14);
- Sistem.RSI(Veriler,14);

**Birinci Kullanım Yöntemi:** Yazdığınız formüldeki 14 Periyotlu RSI indikatörü, formülü hangi hissenin/sembolün kaç dakikalık grafiğine uyguladığınızı kendisi algılar ve o hissenin o dk'lık priyodu için RSI hesaplar. En yaygın kullanım şekli bu şekildedir. Veri olarak üzerine uygulandığı grafik barlarının KAPANIŞ değerlerini kullanır.

**İkinci Kullanım yöntemi(LİSTE):** Kullanıcı tarafından, RSI hesabında kullanılması istenen verinin barların kapanış değeri dışında bir veri listesi olması durumunda kullanılır. Örneğin bir başka indikatörün RSI'ı hesaplatılacaksa bu yöntem kullanılır. Mesela Momentum indikatörünün RSI'ı hesaplatılacaksa Sistem.RSI(Momentum, 14) şeklinde yazılır.

<u>Üçüncü Kullanım Yöntemi(VERİLER)</u>: Bu kullanım yönteminin temelde üç iki ihtiyaç nedeniyle kullanımı söz konusu olur;

a- Formülü örneğin 5 dakikalık grafiğe uygulayacaksınız ama aynı sembolün başka bir (mesela 60 dakikalık) periyodunun RSI değerini kullanmak istiyorsanız kullanılır.

```
var SaatlikVeriler = Sistem.GrafikVerileriniOku(Sistem.Sembol, "60");
var SaatlikRSI = Sistem.RSI(SaatlikVeriler, 14);
```

 b- Formülü örneğin GARAN hisse senedinin grafiği için yazıyorsunuz ama örneğin XU100 endeksinin veya başka bir hissenin RSI değerini kullanmak istiyorsanız kullanılır;

```
var Sembol2 = "IMKBH'HALKB";
var Veriler2 = Sistem.GrafikVerileriniOku(Sembol, "60");
var RSI_HALKB = Sistem.RSI(Veriler2, 14);
```

- AccumulationDistribution - Sistem. AccumulationDistribution

Accumulation Distribution Index, kapanışla günün düşüğü farkından, günün yükseği ile kapanış farkının çıkarılarak, günün yükseği ile günün düşüğü farkına bölünmesi ile elde edilen değerin hacimle çarpılmasıyla elde edilen bir indikatördür. Barların Yüksek ve Düşük Değerleri ve Hacim değerleri gerektiği için, bir liste için hesaplanamaz. (Hesaplanması için HACİM verisi gerektirdiğinden HİSSE veya VIOP için en az DÜZEY1+ / 1 Kademe Derinlik lisansı olmalıdır). Ayrıca parametre almaz. Aşağıdaki gibi 2 yazım şekli vardır;

```
Sistem.AccumulationDistribution();
Sistem.AccumulationDistribution(Veriler);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var X = Sistem.AccumulationDistribution();
Sistem.Cizgiler[0].Deger = X;
```

- ADR - Sistem. ADR(14)

ADR (Average Directional Rating) indikatörünü çağırır. Hesaplanmasında barların Yüksek ve Düşük Değerleri gerektiği için, bir liste için hesaplanamaz. O yüzden LİSTE kullanım şekli yoktur. 1 Adet parametre alır (varsayılan 14). Aşağıdaki gibi 2 yazım şekli vardır;

Sistem.ADR(14); Sistem.ADR(Veriler);

Örnek: İndikatörü çağırıp çizdirmek

var X = Sistem.ADR(14); Sistem.Cizgiler[0].Deger = X;

- ADX - Sistem. ADX(14)

ADX (Average Directional Index) indikatörünü çağırır. 1 Adet parametre alır (varsayılan 14). Aşağıdaki gibi 3 yazım şekli vardır;

```
Sistem.ADX(14);
Sistem.ADX(Liste, 14);
Sistem.ADR(Veriler);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var X = Sistem.ADR(14);
Sistem.Cizgiler[0].Deger = X;
```

```
- Aktif Viop Kontrat - Sistem. AktifViopKontrat
```

VIOP için Sistem kullanan yatırımcılar her 2 ayda bir vadesi sona erip bir başka vade için yeniden açılan XU030 endeks vadeli işlem sözleşmesi için, her zaman en yakın vadeyi ifade eden VIP-X030 kodunu kullanmayı tercih ederler. Bu sayede sayfasına, sistemlerine VIP-X030 sembolü yazanlar, her 2 ayda bir sembol kodunu değiştirmekle uğraşmazlar. Fakat borsaya emir gönderen **robotlar için** mutlaka borsada işlem gören orijinal endeks vadeli işlem sözleşmesinin kodunu yazmak gerekir. Bu nedenle eklenmiş olan Aktif Viop Kontrat fonksiyonu her zaman o an için en yakın vadeli endeks30 vadeli işlem sözleşmesinin orijinal kodunu veren bu komutu kullanırlar.

Şu an en yakın vadeli endeks vadeli işlem sözleşmesinin orijinal kodunu ekrana mesaj penceresi açıp gösteren örnek kullanım kodu;

```
var Sembol = Sistem.AktifViopKontrat;
Sistem.Mesaj(Sembol );
```

var Sembol = Sistem.AktifViopKontrat; Sistem.Mesaj(Sembol );

🖳 iDeal Mesaj 25.05.2020 18:08:48

VIP'F\_XU0300620



C# Programlama dilini iyi düzeyde bilen kullanıcılar için eklenmiştir. Yazılan bir formül (veya Lib.Cs yada User.DLL içerisinden çağrılan bir formül) tyr/catch içerisine alınmışsa ve hata üretmişse (catch içine düşmüşse), hatanın detayını (Exception Error) görmek için kullanılır.

### Örnek Kullanım:

```
try
{
  var C = Sistem.GrafikFiyatSec("Kapanis");
  var RSI = Sistem.RSI(14):
  var MAV = Sistem.MA(6, "Simple", "Kapanis");
  var MACD = Sistem.MACD(12, 26);
  var SonYon = "";
  for (int i = 4; i < Sistem.BarSayisi; i++)
  {
    if (RSI[i] > 70f && MACD[i] > 0.10f && C[i] > MAV[i] && SonYon != "A") // AL
    {
      Sistem.Yon[i] = "A";
      SonYon = Sistem.Yon[i];
    }
    else if (RSI[i] <60f && MACD[i] < -0.10f && C[i] < MAV[i] && SonYon != "S") // SAT
    {
      Sistem.Yon[i] = "S";
      SonYon = Sistem.Yon[i];
    }
  }
  Sistem.Cizgiler[0].Deger = RSI; //Panel 2
  Sistem.Cizgiler[1].Deger = MACD; //Panel 3
  Sistem.Cizgiler[2].Deger = MAV; //Panel 1
}
catch (Exception error) //hata olursa ekrana basması için
{
  string errorline = "r^n + "r^n +
    DateTime.Now.ToString("dd/MM/yyyy HH:mm:ss") + "\r\n" + "\r\n" +
    "Message : " + error.Message + "\r\n" + "\r\n" +
    "Source : " + error.Source + "\r\n" + "\r\n" +
    "StackTrace : " + error.StackTrace + "\r\n";
    Sistem.AlgoAciklama = errorline;
}
```

#### - Alış Fiyat - Sistem. AlisFiyat(Sembol)

Bir sembolün o an ki **en iyi alış fiyatını** okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) en iyi alış fiyatı okutulmak istenen sembol yazılır.

Not: iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir. Örneğin USDTYR için FX'USDTRY şeklinde yazılır. Bir sembolün PİYASA kodunun ne olduğu, o sembolü sayfanıza yazarken @ işareti yanında gösterilir.)

Örnek kullanım şekli aşağıdaki gibidir.

```
var Sembol = "IMKBH'GARAN";
```



var X = Sistem.AlisFiyat(Sembol); Sistem.Mesaj(X.ToString());

var Sembol = "IMKBH'GARAN"; var X = Sistem.AlisFiyat(Sembol); Sistem.Mesaj(X.ToString());

🖷 iDeal Mesaj 25.05.2020 18:25:18

7.54

- Alış Lot - Sistem. AlisLot(Sembol)

Bir sembolün o an ki **en iyi alış fiyatında bekleyen lot miktarını** okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) en iyi alış fiyatında bekleyen lot miktarı okutulmak istenen sembol yazılır.

var Sembol = "IMKBH'GARAN"; var X = Sistem.AlisLot(Sembol); Sistem.Mesaj(X.ToString());

× 🗧 GAR	AN.Y1	DGKT	АВКН	-		
Tavan <b>8.36</b> Taban <b>6.84</b>		Yks 7.60 Dşk 7.51	ÖncK 7.60 Lot 66,406,8	89	Frk% -0.79 Frk -0.06	Dng.F 7.54 Aort 7.544
A.Saat	A.E	A.Lot	Alış	Satış	S.Lot	S.E
18:09:58 18:09:54 18:09:59 18:09:27 18:09:32 18:09:42 18:09:14 18:05:25 18:05:25 18:08:02	133 233 516 668 1243 256 232 155 98 237 3771	634,622 939,658 1,023,270 1,930,382 399,452 286,175 296,939 296,423 331,623 6,653,524	7.54 7.53 7.52 7.51 7.50 7.49 7.48 7.47 7.46 7.45 7.50	7.55 7.56 7.57 7.58 7.59 7.60 7.61 7.62 7.63 7.64 7.59	250,245 551,894 198,248 134,645 434,535 611,579 576,313 629,574 230,695 85,503 3,703,231	21 57 54 57 69 167 56 71 49 50 651
🖳 iDeal I	Mesa	25.05.2020 18:	:41:29			
634622						
var Semb var X = Sistem.M	ol = " <mark>Sistem</mark> esaj(X	'IMKBH'GARAN' 1.AlisLot(Sen (.ToString())	'; nbol); ;;			

#### - Alligator - Sistem. Alligator()

Alligator isimli indikatörü çağırır. Mavi Çizgi Timsahın çenesi, Kırmızı Çizgi Timsahın dişleri ve Yeşil Çizgi Timsahın dudakları olarak kabul edilir. Eğer bu üç çizgi birbirine girerse, Timsah uyuyor demektir, yani piyasa yatay bir kanal içerisindedir. Timsah uzun bir uykudan uyandığında denge çizgileri birbirinden uzaklaşacak ve timsah avlanmaya başlayacaktır. Yeterince avlandıktan sonra ise tekrar uykuya geçecektir. (Denge çizgileri birleşmeye başlar) Bu döneminde kar alma zamanı olduğu varsayılmaktadır.

Eğer timsah uykuda değilse, piyasada aşağı ya da yukarı bir trend var demektir.

-Eğer fiyatlar, timsahın ağzının üstünde ise yukarı yönlü bir trend,

-Eğer fiyatlar, timsahın ağzının altında ise aşağı doğru bir trendin var olduğu kabul edilir.

Mavi, Kırmızı ve Yeşil çizgilerin her biri için ayrı bir fonksiyon vardır.

```
Sistem.Alligator1();
Sistem.Alligator1(Veriler);
Sistem.Alligator2();
Sistem.Alligator2(Veriler);
Sistem.Alligator3();
Sistem.Alligator3(Veriler);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var cizgi1 = Sistem.Alligator1();
var cizgi2 = Sistem.Alligator2();
var cizgi3 = Sistem.Alligator3();
Sistem.Cizgiler[0].Deger = cizgi1;
Sistem.Cizgiler[1].Deger = cizgi2;
Sistem.Cizgiler[2].Deger = cizgi3;
```

- Aroon Up/Down - Sistem. AroonUp(14) / Sistem. AroonDown(14)

Aroon Up ve Aroon Down isimli iki çizgisi olan indikatörü çağırır. Up ve Down çizgileri için ayrı ayrı fonksiyonlar vardır. Piyasada bir Trend var olup olmadığını yorumlamak amaçlı kullanılır. 1 adet parametre alır (varsayılan 14). Bu parametre periyodu boyunca fiyat yeni bir yüksek yaparsa UP olan çizgi 100 değerini alır. Aynı şekilde parametre olarak girilmiş periyot botunca fiyat yeni bir düşük parsa DOWN olan çizgi 100 değerini alır.

Aroon Up ve Aroon Down için çizgilerinin her biri için ayrı bir fonksiyon vardır.

```
Sistem.AroonDown(14);
Sistem.AroonDown(Veriler,14);
Sistem.AroonUp(14);
Sistem.AroonUp(Verileri,14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var Down = Sistem.AroonDown(14);
var Up = Sistem.AroonUp(14);
Sistem.Cizgiler[0].Deger = Down;
Sistem.Cizgiler[1].Deger = Up;
```

### - Aroon Oscilator - Sistem. AroonOsc(14)

Aroon Up ve Aroon Down isimli iki çizgisi olan indikatördeki Up çizgisi ile Down çizgisi arasındaki farkı veren ayrı bir indikatör olan AroonOsc indikatörünü çağırır. Aşağıdaki gibi 2 kullanım şekli vardır

```
Sistem.AroonOsc(14);
Sistem.AroonOsc(Veriler,14);
```

Örnek: İndikatörü çağırıp çizdirmek

var Osc = Sistem.AroonOsc(14); Sistem.Cizgiler[0].Deger = Osc;

- Aşağı Kestiyse - Sistem. AsagiKestiyse

Bir grafik periyodunda bir çizginin bir başka çizgiyi (veya sabit bir yatay seviyeyi/sayıyı) **en son barda** aşağı kesme durumunu verir. Bu fonksiyonun dönüş değeri true veya false (yani evet veya hayır gibi) şeklindedir ve sadece son 2 bardaki değerleri kıyaslar. Geçmişten son bara kadar aşağı kesme durumlarında SAT sinyali üretsin amaçlı bir formülde kullanılmaz. Genellikle Sorgu (Tarama) kodlarında kullanılır. 2 kullanım şekli vardır;

```
Sistem.AsagiKestiyse(Liste1, Liste2))
Sistem.AsagiKestiyse(Liste, Say1))
```

Örnek1: 5'lik hareketli ortalaması 22'lik hareketli ortalamasını aşağı kesen senetleri tarayan sorgu/tarama kodu

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var Mov1 = Sistem.MA(C, "Simple", 5);
var Mov2 = Sistem.MA(C, "Simple", 22);
Sistem.SorguBaslik[0] = "Mov1";
Sistem.SorguBaslik[1] = "Mov2";
if (Sistem.AsagiKestiyse(Mov1, Mov2))
{
Sistem.SorguDeger[0] = Mov1[Sistem.BarSayisi-1];
Sistem.SorguDeger[1] = Mov2[Sistem.BarSayisi-1];
Sistem.SorguAciklama = "Aşağı Kesti";
Sistem.SorguEkle();
}
```

Örnek2: MACD indikatörü 0 seviyesi aşağı kesenleri sorgula/tara kodu

```
var MACD = Sistem.MACD(12,26);
Sistem.SorguBaslik[0] = "MACD";
if (Sistem.AsagiKestiyse(MACD, 0))
{
   Sistem.SorguDeger[0] = MACD[Sistem.BarSayisi-1];
   Sistem.SorguAciklama = "Aşağı Kesti";
   Sistem.SorguEkle();
}
```

55 105 155 V 1 2 4 V 5 8 10 15 20 30												
No	Sembol	Periyot	Açıklama	Mov1	Mov2							
1	AKBNK	1	Aşağı Kesti	5.5100	5.5050							
2	AKBNK	10	Aşağı Kesti	5.5040	5.4886							
3	AKBNK	30	Aşağı Kesti	5.5000	5.4914							
4	AKBNK	5	Aşağı Kesti	5.5140	5.4923							
5	AKBNK	60	Aşağı Kesti	5.5000	5.5014							
6	AKBNK	G	Aşağı Kesti	5.4840	5.6655							
7	AKBNK	н	Aşağı Kesti	5.6300	6.9064							
8	ARCLK	1	Aşağı Kesti	15.5120	15.5050							
9	ARCLK	10	Aşağı Kesti	15.5040	15.4914							
10	ARCLK	30	Aşağı Kesti	15.5020	15.5468							
11	ARCLK	5	Aşağı Kesti	15.5080	15.4977							
12	ARCLK	60	Aşağı Kesti	15.5040	15.4932							
13	ARCLK	G	Aşağı Kesti	15.3060	15.5596							
14	ARCLK	н	Aşağı Kesti	15.4840	17.7173							
15	ASELS	1	Aşağı Kesti	29.2840	29.3100							
16	ASELS	10	Aşağı Kesti	29.3160	29.3882							
17	ASELS	30	Aşağı Kesti	29.3480	29.4500							

ATR (Average True Range) - Sistem. AverageTrueRange(14)

ATR olarak bilinen ve uzun ismi Average True Range indikatörünü çağırır. 1 Adet paramatere alır ve varsayılan olarak 14 değeri kullanılır. Aşağıdaki gibi 3 kullanım şekli vardır

```
Sistem.AverageTrueRange(14);
Sistem.AverageTrueRange(Liste,14);
Sistem.AverageTrueRange(Veriler,14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var ATR = Sistem. AverageTrueRange (14);
Sistem.Cizgiler[0].Deger = ATR;
```

- Awesome Oscilator - Sistem. AwesomeOsc(5, 34)

ATR olarak bilinen ve uzun ismi Average True Range indikatörünü çağırır. Aşağıdaki gibi 2 kullanım şekli vardır

```
Sistem. AwesomeOsc(5, 34);
Sistem. AwesomeOsc(Veriler, 5, 34);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var Awe = Sistem. AwesomeOsc(5, 34);
Sistem.Cizgiler[1].Deger = Awe;
```

- BaglantiVar - Sistem. BaglantiVar

iDeal Veri Terminalinin yayın sunucularına bağlı durumda olup olmadığının kontrolü için kullanılır.

Sistem.BaglantiVar; "true" veya "false" şeklinde değer döner.

### Örnek: Kullanım

```
if (Sistem.BaglantiVar == "true")
{
    Sistem.Mesaj("Yayın var");
}
else
    Sistem.Mesaj("Yayın yok");
```

- BarCiz - Sistem. BarCiz

Grafiklerde indikatör bölgesine çizdirilen çizgilerin, tıpkı fiyat bilgilerinden elde edilen BARLAR gibi (OHLC) değerlerinin hepsini birden gösterecek şekilde ve tipte BAR ÇİZDİRMEK için kullanılır. Örneğin bir indikatörü Açılış, Kapanış, Yüksek ve Düşük değerlerine göre ayrı ayrı hesaplanıp bir BAR gibi çizdirilmesi amaçlı kullanılabilir. Aşağıdaki şekilde yazılır;

Sistem. BarCiz(Panel, BarTip, AcilisListe, YuksekListe, DusukListe, KapanisListe, YukselenRenk, DusenRenk)

**Panel:** 2-3-4 gibi numaralar girilir. Örneğin panel 2 ise, barlar grafiğin altındaki panelde çizilir. **Bar Tipi** 0 ise OHLC, 1 ise MUM barlar, 2 ise HEIKINASHI barlar çizilir.

**AcilisListe:** Bir bar çizdirebilmek için gereken 4 veriden biri Açılış verisidir. Çizdirilecek barların Open (Açılış) değerlerini hangi veri listesinden (bir indikatör mesela) alacağımızı belirtiriz.

**YuksekListe:** Bir bar çizdirebilmek için gereken 4 veriden biri Yüksek verisidir. Çizdirilecek barların High (Yüksek) değerlerini hangi veri listesinden (bir indikatör mesela) alacağımızı belirtiriz.

**DusukListe:** Bir bar çizdirebilmek için gereken 4 veriden biri Düşük verisidir. Çizdirilecek barların Low (Düşük) değerlerini hangi veri listesinden (bir indikatör mesela) alacağımızı belirtiriz.

**KapanisListe:** Bir bar çizdirebilmek için gereken 4 veriden biri Kapanış verisidir. Çizdirilecek barların Close (Kapanış) değerlerini hangi veri listesinden (bir indikatör mesela) alacağımızı belirtiriz.

**YukselenRenk:** Çizilen barların yüksek/düşük renk gösterimi yapılması durumunda, önceki bara göre <u>yüksek olan barın hangi renk olacağı belirtilir</u>. Color.Green şeklinde de belirtilebilir (Color. dan sonra ilk harfi büyük İngilizce renk ismi) Sistem.Renk(0,255,0,0) şeklinde RGB (Kırmızı, yeşil mavi renk oranları) girilerek de renk seçimi yapılabilir.

**DusenRenk:** Çizilen barların yüksek/düşük renk gösterimi yapılması durumunda, önceki bara göre <u>düşük olan barın hangi renk olacağı belirtilir</u>. Color.Red şeklinde de belirtilebilir (Color. dan sonra ilk harfi büyük İngilizce renk ismi) Sistem.Renk(255,50,50,0) şeklinde RGB (Kırmızı, yeşil mavi renk oranları) girilerek de renk seçimi yapılabilir.

Örnek: Grafiğimizin alt paneline, açılış/yüksek/düşük/kapanış değerleri RSI indikatörü olan barlar çizdiren kod (sembolün kendi barlarının açılışlarından hesaplanan RSI, yeni barımızın açılış listesi, sembolün kendi barlarının kapanışlarından hesaplanan RSI, yeni barımızın kapanış listesi, sembolün kendi barlarının yüksek değerinden hesaplanan RSI, yeni barımızın yüksek listesi ve sembolün kendi barlarının düşük değerlerinden hesaplanan RSI, yeni barımızın düşük listesi'dir. Dileyen kullanıcılar,

çılış listesi olarak Mov50, yüksek listesi olarak Mov10, düşük listesi olarak mov100, kapanış listesi olarak mMov200 gibi ayrı indikatörler de kullanabilir.

```
var 0 = Sistem.GrafikFiyatSec("Acilis");
var H = Sistem.GrafikFiyatSec("Yuksek");
var L = Sistem.GrafikFiyatSec("Dusuk");
var C = Sistem.GrafikFiyatSec("Kapanis");
var RSIOpen = Sistem.RSI(0, 14);
```

```
var RSIClose = Sistem.RSI(C, 14);
var RSIHigh = Sistem.RSI(H, 14);
var RSILow = Sistem.RSI(L, 14);
```

// bartip 0:OHLC 1:Candle 2:HeikinAshi
Sistem.BarCiz(2, 1, RSIOpen, RSIHigh , RSILow , RSIClose, Color.Green, Color.Red);



#### - BarRengi - Sistem. BarRengi

Grafiklerdeki BARLARIN her birine ayrı bir renk verebilme imkanı sunar. Aşağıdaki şekilde yazılır;

Sistem. BarRengi(BarNo, Color.Cyan, 1, 1);

Örnek Kullanım:

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var H = Sistem.GrafikFiyatSec("Yuksek");
var L = Sistem.GrafikFiyatSec("Dusuk");
```

```
for (int i=1; i < Sistem.BarSayisi; i++)
{
    if(H[i] > H[i-1] && C[i] < (H[i]+ L[i])/2)
    {
        Sistem.BarRengi(i, Color.Cyan, 3, 1); //barın rengini turkuza mavisi/Cyan yap
    }
    if(L[i] < L[i-1] && C[i] > (H[i]+ L[i])/2)
    {
        Sistem.BarRengi(i, Color.White, 3, 1); //barın rengini GRİ YAP
    }
}
```



- BarSayisi - Sistem. BarSayisi

Bir grafikteki toplam bar sayısını verir. Aşağıdaki şekilde yazılır.

```
var x = Sistem.BarSayisi;
Sistem.Mesaj(X.ToString());
```

#### - Bilanco İndikatörleri - Sistem. Bilanco()

Hisse senetlerinin açıklanmış olan bilançolarından bazı önemli kalemler için, grafikte indirkatör olarak çizilebilenlerin hesaplanması yapan Sistem Fonksiyonlarının listesi ve yazılış şekilleri aşağıdaki gibidir.

```
Sistem.BilancoFK();
Sistem.BilancoPDDD();
Sistem.BilancoOzSerm();
Sistem.BilancoOdenmisSerm();
Sistem.BilancoNetKar();
Sistem.BilancoPD();
```

Örnek: İndikatör olarak bilanço kalemlerini çağırıp çizdirmek



var FK = Sistem.BilancoFK(); var PDDD = Sistem.BilancoPDDD(); var OzSermaye = Sistem.BilancoOzSerm(); var OdSermaye = Sistem.BilancoOdenmisSerm(); var NetKar = Sistem.BilancoNetKar(); var PiyDeg = Sistem.BilancoPD();

Sistem.Cizgiler[0].Deger = FK; Sistem.Cizgiler[1].Deger = PDDD; Sistem.Cizgiler[2].Deger = OzSermaye; Sistem.Cizgiler[3].Deger = OdSermaye; Sistem.Cizgiler[4].Deger = NetKar; Sistem.Cizgiler[5].Deger = PiyDeg;



#### - BistHesapOku - Sistem. BistHesapOku()

Portföy penceresine ekleyerek, parola ve şifrenizi de girip LOGIN olduğunuz aracı kurumdaki hesabınıza ait bilgileri okumak için **Sistem.BistHesapOku()** fonksiyonu kullanılır. Sistem kod editörüne Bu komut yazılıp çalıştırıldığı zaman iDeal aracı kurumunuzdaki hesap için (HİSSE TARAFI) aşağıdaki bilgileri kurumunuzdan çeker ve kullanabilmeniz için sunar;

- İşlem Limiti (bir Sayısal değer döner)
- Kullanılabilir Bakiye (bir sayısal değer döner)
- Pozisyonlar (portföydeki senetleri içeren birden fazla elemanı olan bir listedir)
- Bekleyen Emirler (gün içi bekleyen durumunda olan emirlerinizi içeren bir listedir)
- Gerçekleşen Emirler (gün içi gerçekleşmiş emirlerinizi içeren bir listedir)

Hesap Oku komutu yazıldıktan sonra, yukarıdakilerden hangisine erişilmek isteniyorsa, dönüş okutulurken kullanılan değişkenden sonra NOKTA işareti konulup dönüş verilerden ilgili olanı çağrılır. Dönüş değerlerine erişim (Hesap Oku komutu ile dönen veriler listesi) aşağıdaki gibidir;



```
Sistem.BistHesapOku()
Limit = BistHesap.IslemLimit;
Bakiye = BistHesap.Bakiye;
PozList = BistHesap.Pozisyonlar;
BekleyenList = BistHesap.BekleyenEmirler;
GerceklesenList = BistHesap.GerceklesenEmirler;
```

Pozisyonlar, Bekleyen ve Gerçekleşen Emirler birer Liste oldukları için, onların da dönen verilerinin her biri birden çok veri alanı sunar.

**Pozisyonlar** verisi çekildiği zaman okunabilecek bilgi alanları için, pozisyonları tanımladığımız değişkenin hemen devamına NOKTA işareti koyup sunulan bilgi alanlarının ismi yazılarak, sahip olunan hissenin adedi, maliyeti, senet kodu, kar zarar bilgisi ve Son fiyat değeri elde edilebilir. Pozisyonlar Listesinin dönüş değerleri aşağıdaki gibidir;

```
PozList.Symbol;
PozList.Lot;
PozList.Cost;
PozList.Profit;
PozList.LastPrice;
```

**Gerçekleşen Emirler** verisi çekildiği zaman okunabilecek bilgi alanları için, tanımladığımız değişkenin hemen devamına NOKTA işareti koyup sunulan bilgi alanlarının ismi yazılarak, gün içinde verdiğiniz ve GERÇEKLEŞMİŞ olan emirlerinize ait fiyat, emir no, yön, emir süresi, emir tipi, emir tarihi/saati ve emir statusu (durumu)bilgileri elde edilebilir. Gerçekleşen Emirler Listesinin dönüş değerleri aşağıdaki gibidir;

```
GerceklesenList.OrderNo;
GerceklesenList.OrderDate;
GerceklesenList.Symbol;
GerceklesenList.BuySell;
GerceklesenList.Session;
GerceklesenList.OrderType;
GerceklesenList.Price;
GerceklesenList.Status;
```

**Bekleyen Emirler** verisi çekildiği zaman okunabilecek bilgi alanları için, tanımladığımız değişkenin hemen devamına NOKTA işareti koyup sunulan bilgi alanlarının ismi yazılarak, gün içinde verdiğiniz ve BEKLİYOR DURUMDA OLAN olan emirlerinize ait fiyat, emir no, yön, emir süresi, emir tipi, emir tarihi/saati ve emir statusu (durumu)bilgileri elde edilebilir. Gerçekleşen Emirler Listesinin dönüş değerleri aşağıdaki gibidir;

```
BekleyenList.OrderNo;
BekleyenList.OrderDate;
BekleyenList.Symbol;
BekleyenList.BuySell;
BekleyenList.Session;
BekleyenList.OrderType;
BekleyenList.Price;
BekleyenList.Status;
```

Örnek1: Hesabımı oku, sahip olduğum hisseleri ve onlara ait detayları oku, ekrana bir tablo aç ve tabloya hisselerimi, adetlerini ve maliyetlerini yaz. (formülü kaydedip FORMÜL TEST butonuna basarak sonucu görebilirsiniz.)
<u>Not: Bu sonucu görebilmek için iDeal Portföy penceresinde HERSAP EKLE diyerek aracı kurumdaki</u> <u>hesabınızı eklemiş, parola ve/veya şifrenizi girerek hesabınıza login olmuş olmalısınız.</u>

```
var BistHesap = Sistem.BistHesapOku();
if (BistHesap != null)
{
   var Bakiye = BistHesap.Bakiye;
   var IslemLimit = BistHesap.IslemLimit;
   string TabloAd = "HISSE BAKIYE";
   var SutunGenislik = new int[2]{110,80};
   var SutunHizala = new int[2]{0,2};
   var SutunBaslik = new string[2]{"Açıklama","Değer"};
Sistem.Tablo(TabloAd, 200, 300, 250, 400, 2, 50, SutunGenislik, SutunHizala, SutunBaslik);
   Sistem.TabloTemizle(TabloAd);
     var Renk = Color.Black;
      Sistem.TabloYazdir(TabloAd, 0, 0, "Bakiye", Color.Gold, Renk);
      Sistem.TabloYazdir(TabloAd, 1, 0, Bakiye.ToString("0.00"), Color.White, Renk);
      Sistem.TabloYazdir(TabloAd, 0, 1, "İşlem Limiti", Color.Gold, Renk);
      Sistem.TabloYazdir(TabloAd, 1, 1, IslemLimit.ToString("0.00"), Color.White, Renk);
}
```

- BollingerUp/BollingerDown/BollingerMid/BollingerWidth - Sistem. BollingerXXX()

Bollinger indikatörü 3 adet çizgiden oluşan bir indikatördür. Üst, Orta ve Alt Band çizgilerinin her biri ayrı birer fonksiyon ile okutulup çağrılabilir. Ayrıca Bollinger Üst Band ile Bollinger Alt Band arasındaki farkı sunan (bandın genişliği) Bollinger Width de ayrı bir indikatör4 gibi hesaplatılıp çağrılabilir ve kullanılabilir. Bandın sıkıştığı veya daraldığına yönelik analizler için Width çizgisi tek başına çizdirilip yorumlanabilir.

Kullanım şekilleri aşağıdaki gibidir;

```
//ALT BAND
Sistem.BollingerDown("Simple", 20, 2);
Sistem.BollingerDown(Liste,"Simple", 20, 2);
Sistem.BollingerDown(Veriler,"Simple", 20, 2);
//ÜST BAND
Sistem.BollingerUp(Liste,"Simple", 20, 2);
Sistem.BollingerUp(Veriler,"Simple", 20, 2);
//ORTA BAND
Sistem.BollingerMid("Simple", 20, 2);
Sistem.BollingerMid(Liste,"Simple", 20, 2);
Sistem.BollingerMid(Liste,"Simple", 20, 2);
//BAND GENİŞLİĞİ
Sistem.BollingerWidth(20, 2);
Sistem.BollingerWidth(Veriler,20, 2);
```

Örnek: Fiyatın, Bollinger Orta bandını yukarı kesip %1 de üstüne çıktığı senetleri tarayan sorgu kodu (kaydedilip sorgu isimli pencereden çalıştırılabilir)

```
Sistem.SorguBaslik[0] = "Fiyat";
Sistem.SorguBaslik[1] = "Bol.Orta";
var C = Sistem.GrafikFiyatSec("Kapanis");
var BolOrta = Sistem.BollingerMid(C,"Exp", 20, 1.8);
var sonbar = Sistem.BarSayisi-1;
```

```
if (C[sonbar-1] < BolOrta[sonbar-1] && 1.01F * BolOrta[sonbar] > C[sonbar])
{
    Sistem.SorguDeger[0] = C[sonbar];
    Sistem.SorguDeger[1] = BolOrta[sonbar];
    Sistem.SorguAciklama = "Yukarı Kesti";
    Sistem.SorguEkle();
}
```

- Chaikin Money Flow (CMF) - Sistem. ChaikinMoneyFlow(14)

Chaikin Money Flow (CMF) olarak bilinen indikatörünü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 14 kullanılır. **Hesaplamasında HACİM verisi kullandığından En Az DÜZEY1+ lisans gerektirir**. Aksi durumda hacim içeren barlar gecikmeli olarak kendini günceller. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.ChaikinMoneyFlow(14);
Sistem.ChaikinMoneyFlow(Veriler,14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var CMF = Sistem. ChaikinMoneyFlow (14);
Sistem.Cizgiler[0].Deger = CMF;
```

- Chaikin Osc - Sistem. ChaikinOsc(14)

Chaikin Oscilator olarak bilinen indikatörünü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.ChaikinMoneyOsc(14);
Sistem.ChaikinMoneyOsc(Veriler,14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var CM = Sistem. ChaikinMoneyOsc(14);
Sistem.Cizgiler[0].Deger = CM;
```

- Chaikin Volatility - Sistem. ChaikinVolatility(10,10)

Chaikin Volatility olarak bilinen indikatörünü çağırır. 2 adet parametre (periyot) alır ve varsayılan değer olarak 10-10 kullanılır. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem. ChaikinVolatility(10, 10);
Sistem. ChaikinVolatility(Veriler,10,10);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var CV = Sistem. ChaikinVolatility(10,10);
Sistem.Cizgiler[0].Deger = CV;
```

- Chande Momentum - Sistem. ChandeMomentum(20)

Chande Momentum olarak bilinen indikatörünü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 20 kullanılır. Aşağıdaki gibi 3 kullanım şekli vardır;

```
Sistem. ChandeMomentum(20);
Sistem. ChandeMomentum(Liste,20);
```

Sistem. ChandeMomentum(Veriler,20);

Örnek: İndikatörü çağırıp çizdirmek

```
var Cha = Sistem. ChandeMomentum(20);
Sistem.Cizgiler[0].Deger = Cha;
```

- <u>CizgiCiz - Sistem. CizgiCiz(panel, bar1, fiyat1, bar2, fiyat2, renk, kalinlik, stil)</u>

Grafiklerde kullanıcı tarafından KOD YAZILARAK veya MOUSE İLE KLİKLENEREK tespit edile iki nokta arasına ÇİZGİ ÇİZDİRMEK için kullanılır. Kullanım alanları oldukça geniştir. Otomatik yükselen düşen trendler, destek ve direnç noktaları, pivot seviyeleri, kullanıcı açısından koşullandırılmış iki dönem arasını işaretleme gibi çok çeşitlki amaçlar için kullanılabilir. Aşağıdaki şekilde yazılır;

Sistem.CizgiCiz(panel, bar1, fiyat1, bar2, fiyat2, renk, kalinlik, stil)

• Panel = Çizgi barların olduğu bölgede çizilecekse 1 değeri verilmelidir.

Aşağıdaki indikatör çizim bölgelerinde çizdirilecekse 2 veya diğer panel numaraları kullanılır.

- Bar1 ve Bar2 bilgileri çizginin başlayacağı ve biteceğiz barların numarasıdır.Elle sabit bar nosu da girilebilir. (500 ve 1500 gibi)
- Fiyat1 ve Fiyat2 çizgilerin başladığı ve bittiği barların HANGİ NOKTASINDAN çizilmelidir (barın en tepesinden, dibinden, ortasından veya hesaplatılan bir fiyat seviyesinden)
- Renk = Color.Red gibi noktadan sonra İngilizce renk ismi yazılarak çizgi rengi atanabilir.
- Kalinlik = 1,2,3,4,5 değerleri verilebilir
- Stil değerine 1-2-3-4-5 değerleri verilerek DÜZ, KESİK, NOKTA, DİKEY ve YASSI tipte çizgi çizdirtilebilir.

Örnek Kullanım: Mouse ile tıklanan yerden başla, devam eden 1200 bar sonrasına kadarki bölgenin EN YÜKSEK ve EN DÜŞÜK seviyesine ve ikisinin tam orta seviyesine yatay çizgi çiz Not: Mouse ile tıklanan bölgeyi algılayıp formül içinde kullanma kısmı için SELECTBARNO komutu açıklamasını inceleyiniz.

```
var V = Sistem.GrafikVerileri;
int basla = Sistem.SelectBarNo; //mouse ile kliklenen bardan itibaren çizmeye başla
var bitis = Sistem.SelectBarNo + 1200; //kliklenen yerden sonra 1200 bar içindeki bölgeyi çiz
if (bitis > Sistem.BarSayisi-1) bitis = Sistem.BarSayisi-1; //1200 bar yoksa son bara kadarv  çiz
float yuksek = V[basla].High;
float dusuk = V[basla].Low;
for (int i=basla; i <= bitis; i++)
{
    yuksek = Math.Max(yuksek,V[i].High);
    dusuk = Math.Min(dusuk,V[i].Low);
}
float pivot = (yuksek + dusuk)/2;
Sistem.CizgiCiz(1, basla, yuksek, bitis, yuksek, Color.Green, 3, 1);
Sistem.CizgiCiz(1, basla, dusuk, bitis, dusuk, Color.Red, 3, 3);
Sistem.CizgiCiz(1, basla, pivot, bitis, pivot, Color.White, 3, 2);
```



#### - Cizgiler - Sistem.Cizgiler[x].Deger

IDEAL sistem modülünde yazılan bir formülde, grafik pencereleri üzerine bir çizgi çizilmesi gerektiğinde **Sistem.Cizgiler** fonksiyonu kullanılır. Kendi özel indikatörünüz, bir veri listesi, sabit bir değer listesi veya bir indikatör bu fonksiyonla ekrana çizdirilir.

Bir çizgi demek, zaman birimlerinde bir değere sahip olan bir veri listesi demektir. O yüzden tek bir sabit değere bir yatay çizgi bile çizdirilmek istense, tüm elemanları o sabit değer olan bir liste tanımlamak gerekir.

IDEAL sistem modülü en fazla 50 adet çizgiye izin vermektedir. Gelişmiş sistem paneli (kodlamanın yapıldığı ana pencerenin sağ üst kısmı), çizgilerin pek çok özelliğine kod yazmadan müdahale imkânı sunmaktadır.

Çizginin alacağı değerlerin ne olacağı dışındaki tüm özellikler, kod yazmaksızın panel üzerinden seçimlerle belirlenebilir.

Bir çizgi için aşağıdaki özellikler bulunmaktadır

- Deger (zorunlu)
- Aciklama
- ActiveBool (koddan yazılmazsa, x numaralı çizginin panelden AKTİF edilmesi zorunlu)
- Stil (aksi belirtilmezse DÜZ çizgi stili kullanılır)
- Kalınlık (Aksi belirtilmezse kalınlık 1 olarak kabul edilir)
- Renk (Panelde varsayılan olarak her çizginin tarafımızdan atanmış bir rengi vardır. Kullanıcı istediği rengi seçebilir)
- Panel (Grafiğin kendisinin panel numarası SIFIRDIR. Grafiğin alt kısmına (indikatör bölgesine) atılan her alan yeni bir paneldir. Grafiğin hemen altındaki ilk bölgenin panel numarası "1" dir ve maksimum panel sayısı 10'dur.

\*\*\* Her ÇİZGİ bir numaraya sahiptir ve tanımlanırken çizginin numarası mutlaka (köşeli parantez içinde) belirtilmelidir. **İLK ÇİZGİ NUMARASI SIFIRDIR.** 

\*\*\*Cizgi fonksiyonu kullanılırken Sistem.Cizgiler[CizgiNumarası].OZELLİK şeklinde yazılmalıdır. Özellik olarak yukarıda listelenen ifadeler kullanılabilir.

\*\*\*KOD yazarken **Sistem.Cizgiler[0].Deger** satırını yazmak yeterli ve **ZORUNLUDUR.** (aktif etme, panel numarası, renk, stil, kalınlık PANEL üzerinden seçilebilir.

Örnek Kullanım: Alt indikatör paneline (Panel No = 2) RSI indikatörünü ve aynı zamanda 30 ve 70 seviyelerine yatay çizgileri Kod yazarak ve tüm çizgi özelliklerini koddan atama yaparak çizdirmek.

```
var RSI = Sistem.RSI(14);
Sistem.Cizgiler[0].Deger = RSI;
Sistem.Cizgiler[0].Aciklama = "RSI";
Sistem.Cizgiler[0].ActiveBool = true;
Sistem.Cizgiler[0].Panel = 2;
Sistem.Cizgiler[0].Renk = Color.Gold;
Sistem.Cizgiler[0].Stil = 1; //1:Düz, 2:Nokta, 3:Kesik, 4:Yassi
Sistem.Cizgiler[0].Kalinlik = 2;
Sistem.Cizgiler[1].Deger = Sistem.Liste(70);
Sistem.Cizgiler[1].ActiveBool = true;
Sistem.Cizgiler[1].Stil = 3; //1:Düz, 2:Nokta, 3:Kesik, 4:Yassi
Sistem.Cizgiler[1].Kalinlik = 2;
Sistem.Cizgiler[1].Panel = 2;
Sistem.Cizgiler[1].Renk = Color.White;
Sistem.Cizgiler[1].Aciklama = "ÜstSeviye";
Sistem.Cizgiler[2].Deger = Sistem.Liste(30);
Sistem.Cizgiler[2].ActiveBool = true;
Sistem.Cizgiler[2].Panel = 2;
Sistem.Cizgiler[2].Renk = Color.Lime;
Sistem.Cizgiler[2].Stil = 3; //1:Düz, 2:Nokta, 3:Kesik, 4:Yassi
Sistem.Cizgiler[2].Kalinlik = 2;
Sistem.Cizgiler[2].Aciklama = "AltSeviye";
```

Yukarıdaki formül kaydedildiği zaman, kod yazma panelinin sağ üst kısmındaki yazı/renk/tip seçimleri aşağıdaki gibi olur.

ลเ	No	Acıklama	Aktii	Panel	Renk	Kalinlik	Stil	t
Ξ	0	Kapanış Fiyatları		2	•	4	1 : Düz	L
	1	ÜstSeviye		2	•	2	3 : Kesik	I
	2	AltSeviye		2	•	2	3 : Kesik	L
	3			-	-	-	1.002	4
					-		·	

**NOT:** Özellikle mail yoluyla iDeal Data Teknik ekibinden formül desteği alındığında, kullanıcılarımızın en çok yaptığı hata çizgileri aktif etmemektir.

Maille gelen bir formül kaydedilip grafiğe uyguladığında, ekrana bir çizgi çizilmesi gerekiyorsa (formül içinde genelde de en altta Sistem Çizgiler satırı varsa) ilgili çizgi numaralarının üstteki fotoda gösterilen panelde, başlarındaki kutucuların işaretlenerek aktif edilmesi, çizgi grafiğin alt kısmına çizilmesi gerekiyorsa panel numarasının 2 (veya 3-4-5) olarak kaydedilmesi gerekir.



- <u>CCI – Commodity Channel Index - Sistem. CommodityChannelIndex(14)</u>

CCI olarak da bilinen Commodity Cahannel Index indikatörünü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi 2 kullanım şekli vardır;

Sistem. CommodityChannelIndex(14); Sistem. CommodityChannelIndex(Veriler,14);

Örnek: İndikatörü çağırıp çizdirmek

```
var CCI = Sistem. CommodityChannelIndex (14);
Sistem.Cizgiler[0].Deger = CCI;
```

- Debug - Sistem. Debug()

IDEAL Sistem modülünde bir formül yazılırken, kodun herhangi bir noktasında DEBUG fonksiyonu kullanılarak kodun o noktasında bir sorun olup olmadığı, kodda bir sorun varsa, sorunun ne olduğu daha rahat tespit edilebilir.

Debug fonksiyonu, IDEAL Sistem panelinin sağ üst bölgesindeki çizgilere ait tercih bölgesinin üzerini kaplayan bir boş pencere açar ve orada, debug edilen satırdaki bulgular gösterilir.

Sistem.Debug() fonksiyonu çalışma mantığı bakımından MESAJ fonksiyonuna çok benzer. Tek farkları, ekrana bir mesaj kutusu çıkarılması yerine, sonuç değerlerinin, panelde gösterilmesidir.

Bu özelliği nedeniyle, hata bulmak ve değişebilen bir sonucu gözlemlemek amacıyla mesajdan daha yararlıdır.

Kod içinde birden fazla mesaj fonksiyonu varsa, satır olarak en aşağıda olan mesaj fonksiyonunun sonucu ekranda görülür.

Ama birden fazla satıra debug konulmuşsa, debug ekranında bunları hepsi alt alt aynı anda gösterilir.

(kodda en altta yer alan debug satırının sonuç değeri, debug panelinde ilk sırada gösterilir.)

Debug butonuna basılara DEBUG alanı aktif edilir. Daha sonra FORMÜL TEST butonuna basılınca, Tüm debug satırlarının sonuçları panelde gösterilir.

Örnek: Aşağıdaki kod yazılıp kaydedilir ve FORMÜL TEST butonuna basılırsa, Panelin üst kısmındaki Debug penceresinde izlemek istediğimiz bilgileri görebiliriz;

```
var Sembol = "VIP'F_XU0300620";
var Veriler = Sistem.GrafikVerileriniOku(Sembol, "20");
```

```
Sistem.Debug("SON BARIN KAPANIS FIYATI = " + Veriler[Veriler.Count-1].Close.ToString());
Sistem.Debug("537. BARIN TARİH VE SAATİ = " + Veriler[537].Date.ToString());
Sistem.Debug("450.BARIN SAAT BİLGİSİ = " + Veriler[450].Date.Hour.ToString());
Sistem.Debug("120.BARIN DAKİKA BİLGİSİ = " + Veriler[120].Date.Minute.ToString());
```



#### - DEMA – Sistem. DEMA(5)

DEMA indikatörünü çağırır. 1 adet parametre (periyot) alır. Aşağıdaki gibi 3 kullanım şekli vardır;

```
Sistem.DEMA(5);
Sistem.DEMA(Liste,5);
Sistem.DEMA(Veriler,5);
```

Örnek: DEMA5 değeri DEMA 21 değerisini aşağı keserse SAT, yukarı keserse AL sinyali üreten strateji kod örneği

Not: KESİŞME TARA fonksiyonu detayı için ilgili fonksiyon detayını inceleyiniz.

```
var DEMA1 = Sistem.DEMA(5);
var DEMA2 = Sistem.DEMA(21);
```



Sistem.KesismeTara(DEMA5, DEMA21);

Sistem.Cizgiler[0].Deger = DEMA5; Sistem.Cizgiler[1].Deger = DEMA21;



#### - Demand Index - Sistem.DemandIndex()

Demand Index olarak bilinen indikatörünü çağırır. Hiç parametre almaz ve **Hesaplamasında HACİM verisi kullandığından En Az DÜZEY1+ lisans gerektirir**. Aksi durumda hacim içeren barlar gecikmeli olarak kendini günceller. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.DemandIndex();
Sistem.DemandIndex(Veriler);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var Demand = Sistem.DemandIndex();
Sistem.Cizgiler[0].Deger = Demand;
```

- De Marker - Sistem.DeMarker(13)

De Marker olarak bilinen indikatörünü çağırır. 1 adet parametre alır ve varsayılan parametre olarak 13 kullanılır. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.DeMarker(13);
Sistem.DeMarker(Veriler,13);
```

Örnek: İndikatörü çağırıp çizdirmek



var Marker = Sistem.DeMarker(13); Sistem.Cizgiler[0].Deger = Marker;

- Derinlik Verileri - Sistem.DerinlikVerisiOku(Sembol)

Bir Hisse Senedinin, Varantın, Vadeli Kontratın ve Opsiyonun (lisansınız var ise) DERİNLİK penceresindeki tüm verilerini okumak için kullanılır. Okunan veri biz **dizi** şeklindedir. Derinlik penceresinin ALIŞ ve SATIŞ tarafı ayrı ayrı ve SATIR SATIR okunup hafızaya alınması ve okunan bu verilerin formüllerde kullanılması mümkündür.

Bu fonksiyonla okutulan Veri Listesinin her bir elemanı şu şekilde çağırılır; VeriListesi.Bids/Asks[DerinlikSatırNo].Eleman

Örnek1: VIOP Aktif kontratının Derinlik penceresindeki ilk satıra (hem alış hem satış tarafı) ait bazı verileri ekrana mesaj olarak almak

var Sembol = "VIP'VIP-X030";

var Derinlik = Sistem.DerinlikVerisiOku(Sembol);

var AlisFiyatKademe0 = Derinlik.Bids[0].Price;

var AlisLotKademe0 = Derinlik.Bids[0].Size;

var AlisEmirSayisi = Derinlik.Bids[0].OrderCount;

var SatisFiyatKademe0 = Derinlik.Asks[0].Price;

var SatisLotKademe0 = Derinlik.Asks[0].Size;

var SatisEmirSayisi = Derinlik.Asks[0].OrderCount;

var SatisEmirSaati = Derinlik.Asks[0].Time;

Sistem.Mesaj("Alış Fiyatı =" + AlisFiyatKademe0.ToString() + "\r\n" + "Alış Lot = " + SatisEmirSaati.ToString() + "\r\n" + "Alış Emir = " + AlisEmirSayisi.ToString()+ "\r\n" + "Satış Emir Sayısı = " + SatisEmirSayisi.ToString());

Bir sembolün derinlik penceresine ait veriler aşağıdaki şekilde erişilebilir durumdadır

var Veriler = Sistem.DerinlikVerisiOku(Sembol)										
	ALIŞ TAR/	AFI		SATIŞ TARAFI						
Saat Emir Sayısı Lot Fiyat			Fiyat	Lot	Emir Sayısı	Saat				
Bids[0].Time;	Bids[0].OrderCount;	Bids[0].Size;	Bids[0].Price;	Ask[0].Price;	Asks[0].Size;	Asks[0].OrderCount;	Asks[0].Time;			
Bids[1].Time;	Bids[1].OrderCount;	Bids[1].Size;	Bids[1].Price;	Ask[1].Price;	Asks[1].Size;	Asks[1].OrderCount;	Asks[1].Time;			
Bids[2].Time;	Bids[2].OrderCount;	Bids[2].Size;	Bids[2].Price;	Ask[2].Price;	Asks[2].Size;	Asks[2].OrderCount;	Asks[2].Time;			
Bids[3].Time;	Bids[3].OrderCount;	Bids[3].Size;	Bids[3].Price;	Ask[3].Price;	Asks[3].Size;	Asks[3].OrderCount;	Asks[3].Time;			
Bids[4].Time;	Bids[4].OrderCount;	Bids[4].Size;	Bids[4].Price;	Ask[4].Price;	Asks[4].Size;	Asks[4].OrderCount;	Asks[4].Time;			

NOT: DERİNLİK pencereleri altındaki TOPLAM satırındaki verileri okutmak için, derinlik verisi oku dedikten sonra Derinlik.CalculateAverage2(); satırını da yazmak gerekir. Sonra ilgili bilgiye erişilebilir.

Örnek2: DOAS hisse senedine ait derinlik penceresindeki bazı verileri okuyup ekrana MESAJ olarak gösteren kod örneği;

```
var Sembol = "IMKBH'DOAS";
var Derinlik = Sistem.DerinlikVerisiOku(Sembol);
var AlisFiyatKademe0 = Derinlik.Bids[0].Price;
var AlisLotKademe0 = Derinlik.Bids[0].Size;
var AlisEmirSayisi = Derinlik.Bids[0].OrderCount;
var SatisFiyatKademe0 = Derinlik.Asks[0].Price;
var SatisLotKademe0 = Derinlik.Asks[0].Size;
var SatisEmirSayisi = Derinlik.Asks[0].OrderCount;
var SatisEmirSayisi = Derinlik.Asks[0].OrderCount;
var SatisEmirSayisi = Derinlik.Asks[0].Time;
```

NOT: Derinlik penceresinin en altındaki toplam satırlarında gözüken verilere erişilecekse, kodun için aşağıdaki satır (ortalamaları hesapla) eklenmeli. Derinlik.CalculateAverage2();

var Bid = Derinlik.Average2Bid.Size; Sistem.Mesaj(Bid.ToString());

Sistem.Mesaj("Alış Fiyatı =" + AlisFiyatKademe0.ToString() +
"\r\n" + "Alış Lot = " + AlisLotKademe0.ToString("0") +
"\r\n" + "Satış Fiyatı = " + SatisFiyatKademe0.ToString()+
"\r\n" + "Satış Lot = " + SatisLotKademe0.ToString("0")+
"\r\n" + "Saat = " + SatisEmirSaati.ToString()+
"\r\n" + "Toplam Lot = " + Derinlik.Average2Bid.Size.ToString());

× + DOA	S.Y2 D	GKTA	в к н -				_ ×
Tavan <b>10.80</b> Taban <b>8.84</b>		Yks <b>10.18</b> Dşk <b>9.65</b>	ÖncK 9.82 Lot 3,740,64	47	Frk% 3.46 Frk 0.34	Dng.Fy 10.16 Aort 9.9160	rt )
A.Saat	A.E	A.Lot	Alış	Satış	S.Lot	S.E	S.Saat
18:09:58 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46	3 3 4 2 2 2 1 2 3 42 42	12,740 5,700 3,700 1,113 200 513 5,100 200 300 50 29,616 224,672	10.16 10.15 10.14 10.13 10.12 10.11 10.10 10.09 10.08 10.07 10.14 9.78	10.17 10.18 10.19 10.20 10.21 10.22 10.23 10.24 10.25 10.26 10.21 10.42	965 13,984 10,995 25,405 23,103 2,531 8,110 110 9,007 8,178 102,388 309,393	1 8 27 5 6 2 15 4 90 58	18:09:57 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46 18:09:46
🖳 iDeal I	Mesaj 🛛	25.05.2020 23:	15:31				×
Alış Fiyatı = 10.16 Alış Lot = 12740 Satış Fiyatı = 10.17 Satış Lot = 965 Saat = 180957 Toplam Lot = 224672							~

Örnek3: Alışta bekleyen lot sayısı, satışta bekleyen lot sayısının 2 katından fazla olan ve aynı zamanda son fiyatı 50'lik ortalamasının üstünde olan senetleri tarayan kod örneği (sorgu):

```
var Derinlik = Sistem.DerinlikVerisiOku(Sistem.Sembol);
var C = Sistem.GrafikFiyatSec("Kapanis");
var Mov = Sistem.MA(C, "Exp", 50);
var Alistoplam = 0;
var Satistoplam = 0;
for (int i = 0; i < 6; i++)
{
   Alistoplam += Derinlik.Bids[i].Size;
   Satistoplam += Derinlik.Asks[i].Size;
}
Sistem.SorguBaslik[0] = "Alış Lot";
Sistem.SorguOndalik[0] = 0;
Sistem.SorguBaslik[1] = "Satış Lot";
Sistem.SorguOndalik[1] = 0;
Sistem.SorguBaslik[2] = "Kapanış";
Sistem.SorguBaslik[3] = "Mov";
```

```
var sonbar = Sistem.BarSayisi-1;
if (Alistoplam > 2 * Satistoplam && C[sonbar] > Mov[sonbar] )
{
   Sistem.SorguDeger[0] = Alistoplam;
   Sistem.SorguDeger[1] = Satistoplam ;
   Sistem.SorguDeger[2] = C[sonbar];
   Sistem.SorguDeger[3] = Mov[sonbar] ;
   Sistem.SorguAciklama = "Filtrem";
   Sistem.SorguEkle();
}
```

- De Trended Price Oscilator - Sistem.DetrendedPriceOscillator(14)

De Trended Price Oscillator indikatörünü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi 3 kullanım şekli vardır;

```
Sistem.DetrendedPriceOscillator(14);
Sistem.DetrendedPriceOscillator(Liste,14);
Sistem.DetrendedPriceOscillator(Veriler,14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var DT = Sistem.DetrendedPriceOscillator(14);
Sistem.Cizgiler[0].Deger = DT;
```

```
- Devre Kesici Listesini Oku - Sistem. DevreKesiciListesiniOku()
```

Bir hisse senedinde borsa sisteminde devre kesici durumu oluştuğu zaman, toplu olarak tüm hisselerdeki durumları da tespit ederek listenin tümünü okumak için kullanılır. Formül alanına aşağıdaki komut yazıldığında DEVRE KESİCİ LİSTESİ okunup belleğe alınmış olur; Sistem. DevreKesici ListesiniOku()

Daha sonra ise bu listenin içindeki bilgi alanlarına erişilebilir. Listeyi çağırdığınızda aşağıdaki bilgileri içeren bir liste gelir.

```
string Sembol = "";
string Saat = "";
string StateCode = "";
```

State Code değeri 55 olan hisseler devre kesici dönemine girmiştir

AYRICA YÜZEYSEL VERİ LİSTESİNİ OKUTARAK BİR TEK HİSSE SENEDİ İÇİN devre kesici durumunun ne olduğu bilgisine erişilebilir.

Not: Yüzeysel Veri Oku Fonksiyonu detayları için ilgili komutun açıklama sayfasını inceleyiniz.

Aşağıdaki örnek kod ARTI hisse senedinin borsadaki o anki durum kodunu veriri. Durum kodu 2 olanlar sürekli müzayede seansında olanları, 55 olanlar devre kesmiş olanları veriri. Tüm durum kodlarının ne olduğu bilgisi için gerek duyulursa ideal algo ekibine (algo@idealdata.com.tr) başvurulabilir.



var Sembol = "IMKBH'ARTI"; var V = Sistem.YuzeyselVeriOku(Sembol); var Durum = V.Durum;

- Dikey Çizgi Ekle - Sistem.DikeyCizgiEkle(BarNo, Color.Yellow, 2, 1)

Grafiklerde kullanıcı tarafından istenilen herhangi bir bara DİKEY ÇİZGHİ çizdirilmesi imkanı verir.

Kullanımda yazım şekli aşağıdaki gibidir;

Sistem.DikeyCizgiEkle(i, Color.Green, 2, 2);

Burada çizginin rengi, kaç numaralı BARA çizginin çizileceği, çizginin kalınlığı ve tipi (düz/kesik) belirtilir.

Örnek: RSI indikatörünün 30 seviyesini aşağıdan yukarı kestiği barlara YEŞİL dikey çizgi ekleyen, RSI indikatörünün 70 seviyesini yukarıdan aşağıya kestiği barlara KIRMIZI dikey çizgi ekleyen kor örneği;

```
var RSI = Sistem.RSI(14);
```

```
for(int i=1 ; i< Sistem.BarSayisi; i++)
{
    if(RSI[i-1] < 30 && RSI[i] >= 30)
        Sistem.DikeyCizgiEkle(i, Color.Green, 2, 2);
    if(RSI[i-1] > 70 && RSI[i] <= 70)
        Sistem.DikeyCizgiEkle(i, Color.Red, 2, 2);
}</pre>
```





#### - Dip - Sistem. Dip(Barsayısı)

Belli bir bar sayısı kadar bölgenin DİP seviyesini gösteren DİP indikatörünü hesaplayan fonksiyondur. Girilen bar sayısı kadar öncesi ve sonrası bölgenin en dip noktasını verir. Kullanım şekli aşağıdaki gibidir;

Sistem.Dip(100);

#### Bkz: Sistem.Zirev(100) aynı zamanda.

#### Örnek1: 50 periyotluk Dip/Zirve indikatörü çizdirme kodu

var Dip = Sistem.Dip(50); var Zirve = Sistem.Zirve(50);

```
Sistem.Cizgiler[0].Deger = Dip;
Sistem.Cizgiler[1].Deger = Zirve;
```



#### DI+ / DI- (PDI/MDI) - Sistem. DirectionalIndicatorMinus/Plus(14)

DI+ ve DI- olarak iki çizgisi olan veya Plus(+) olanın PDI, Minus(-) olanın MDI olarak da adlandırıldığı Directonal Indicator indikatörünü çağıran fonksiyondur. DI+ için bir ve DI- için bir olmak üzere iki ayrı komutu vardır. Kullanım şekilleri aşağıdaki gibidir;

```
Sistem.DirectionalIndicatorMinus(14));
Sistem.DirectionalIndicatorMinus(Veriler,14));
Sistem.DirectionalIndicatorPlus(14));
Sistem.DirectionalIndicatorPlus(Veriler,14));
```



#### Örnek1: 14 periyotluk DI+/DI- indikatörü çizdirme kodu

```
var MDI = Sistem.DirectionalIndicatorMinus(14);
var PDI = Sistem.DirectionalIndicatorPlus(14);
Sistem.Cizgiler[0].Deger = MDI;
Sistem.Cizgiler[1].Deger = PDI;
```



#### - Directional Movement - Sistem. DirectionalMovement(14)

Directonal Movement indikatörünü çağıran fonksiyondur. Tek bir parametre alır ve varsayılan parametre değeri 14'tür. Kullanım şekilleri aşağıdaki gibidir;

```
Sistem.DirectionalMovement(14);
Sistem.DirectionalMovement(Verileri,14);
```

Örnek1: 14 periyotluk Directional Movement indikatörü çizdirme kodu

```
var DM = Sistem.DirectionalMovement(14);
Sistem.Cizgiler[0].Deger = DM;
```

```
- Dolgu Ekle - Sistem. DolguEkle()
```

Bir grafik üzerinde çizdirilen 2 çizgi arasında, bu çizgilerin birbirlerini kestikleri bölgeleri seçilen renklerle boyama amaçlı kullanılır. Kullanım şekli aşağıdaki gibidir;

```
Sistem.DolguEkle(No1, No2, YukselisRenk, DususRenk);
```

Burada No1, dolgu ile boyanması istenen alanı oluşturacak birinci çizgi numarası, No2 ise söz konusu boyalı alanı oluşturacak diğer çizginin numarasıdır. YükelisRenk bilgisi girilen iki çizgiden birincisi ikincisinin üstünde olduğu bölgenin rengini, DususRenk ise birinci çizginin ikinci çizginin altında kaldığı bölgenin boyanacağı bölge rengini belirtir.

Renkler belirtilirken C# dilinin renk tanımlayan kod yazım şekilleri (R-G-B oranları vererek renk atamak vs.) geçerlidir ama en kolayı Color.Red, Color.Blue şeklinde, Color.IngilizceRenkAdı şeklinde tanımlamaktır.

Örnek: MACD indikatörü ile SIFIR seviyesindeki çizgi arasında kalan bölgeleri, MACD üstteyken yeşil, MACD alttayken kırmızı renk ile dolgu yapan kod:

```
var MACD = Sistem.MACD(12,26);
```

```
Sistem.Cizgiler[0].Deger = MACD;
Sistem.Cizgiler[1].Deger = Sistem.Liste(0);
Sistem.DolguEkle(0, 1, Color.Green, Color.Red);
```



#### - Dönem Çevir - Sistem.Donemcevir()

Bir grafikte çalışma yapılan periyottan daha üst bir zaman dilimine ait barlara ait bilgileri, bu üst dönem için çizilen indikatörleri alt periyotlarda kullanmak veya çizdirmek için kullanılır. Kullanım şekli aşağıdaki gibidir;

```
Sistem.DonemCevir(Veriler, UstDonemVerileri, UstDonemCevrilecekData);
```

**Veriler** yazan yere, çalışılan periyot ne ise, o periyoda ait BAR VERİLERİ yazılır. Eğer formülde bu veriler bir değişkene atanmışsa o değişken ismi yazılır. Üst zaman diliminden alınan bilgileri değişik alt zaman periyotlarında kullanacaksak (alt dönem sabit bir periyot değilse) o zaman

Veriler yazan kısma Sistem.GrafikVerileri yazılır. Bu komut, formül hangi zaman periyoduna uygulanmışsa, o zaman diliminin BAR VERİLERİNİ (OHLC tamamını birden) verir.

**UstDonemVerileri** yazan yere, daha üst zaman dilimi olarak hangi grafik periyodu kullanılacaksa o zaman diliminin BAR VERİLERİ yazılır. Örnek: var GUNVERILER = Sistem.GrafikVerileriniOku(Sembol, "G") diyerek GÜNLÜK BAR VERİSİ çağrılır.

**UstDonemCevrilecekData** yazan yere, üst zaman diliminden hesaplanmış ve alt dönemde kullanılacak veya çizdirilecek veri (Fiyat veya indikatör) hangisi ise o yazılır.

Örnek: Günlük, 240 dakikalık, 60 dakikalık ve 30 dakikalık periyotların 50'lik Movlarını 10 dklık grafik üzerine çizdirmek;

```
//üst periyot verilerini oku
var V_GUN = Sistem.GrafikVerileriniOku(Sistem.Sembol,"G");
var V_240 = Sistem.GrafikVerileriniOku(Sistem.Sembol,"240");
var V_60 = Sistem.GrafikVerileriniOku(Sistem.Sembol,"60");
var V 30 = Sistem.GrafikVerileriniOku(Sistem.Sembol,"30");
var CGun = Sistem.GrafikFiyatOku(V_GUN, "Kapanis");
var C240 = Sistem.GrafikFiyatOku(V_240, "Kapanis");
var C60 = Sistem.GrafikFiyatOku(V 60, "Kapanis");
var C30 = Sistem.GrafikFiyatOku(V_30,"Kapanis");
var MovGUN = Sistem.MA(CGun, "Simple", 50);
var Mov240 = Sistem.MA(C240, "Simple", 50);
var Mov60 = Sistem.MA(C60, "Simple", 50);
var Mov30 = Sistem.MA(C30, "Simple", 50);
//Verilerin dönemini çevir
MovGUN = Sistem.DonemCevir(Sistem.GrafikVerileri, V_GUN, MovGUN);
Mov240 = Sistem.DonemCevir(Sistem.GrafikVerileri, V_240 , Mov240);
Mov60 = Sistem.DonemCevir(Sistem.GrafikVerileri, V_60 , Mov60);
Mov30 = Sistem.DonemCevir(Sistem.GrafikVerileri, V_30, Mov30);
//ekranda cizgi olarak göster
Sistem.Cizgiler[0].Deger = MovGUN;
Sistem.Cizgiler[0].Aciklama = "MovGUN" ;
Sistem.Cizgiler[1].Deger = Mov60;
Sistem.Cizgiler[1].Aciklama = "Mov60 Dk" ;
Sistem.Cizgiler[2].Deger = Mov30;
Sistem.Cizgiler[2].Aciklama = "Mov30 Dk" ;
Sistem.Cizgiler[3].Deger = Mov240 ;
Sistem.Cizgiler[3].Aciklama = "Mov240 Dk" ;
```



#### - Dörtgen Çiz - Sistem.DortgenCiz()

Grafik üzerinde kullanıcı tarafından girilen 4 noktayı (4 bar numarası ve 4 fiyat seviyesi) birleştirerek dörtgen çizme amaçlı kullanılır. Özellikle patern/Flama/Formasyon bulma amaçlı kodlar yazan kullanıcılarımız, formasyonu oluşturan barları ve fiyat seviyeleri koddan tespit edip formasyonun olduğunu otomatik olarak dörtgen veya kelebek şeklinde çizdirtebilirler.

Kullanım şekli aşağıdaki gibidir;

DortgenCiz(panel, bar1, fiyat1, bar2, fiyat2, bar3, fiyat3, bar4, fiyat4, renk, kalinlik, dolgu, dolgurenk, nokta, noktarenk);

#### Parantez içindeki giriş bilgileri açıklamaları aşağıdaki gibidir;

- Panel: Dörtgenin grafiğin hangi paneline çizileceği girilir (barların olduğu bölge=1, alt indikatör bölgesi = 2)
- Bar1: Dörtgenin birinci köşesinin denk geldiği bar numarası
- Fiyat1: Dörtgenin birinci köşesinin fiyat skalasına denk gelen seviyesi
- Bar2: Dörtgenin ikinci köşesinin denk geldiği bar numarası
- Fiyat2: Dörtgenin ikinci köşesinin fiyat skalasına denk gelen seviyesi
- Bar3: Dörtgenin üçüncü köşesinin denk geldiği bar numarası
- Fiyat3: Dörtgenin üçüncü köşesinin fiyat skalasına denk gelen seviyesi
- Bar4: Dörtgenin dördüncü köşesinin denk geldiği bar numarası
- Fiyat4: Dörtgenin dördüncü köşesinin fiyat skalasına denk gelen seviyesi
- Renk: Dörtgenin dış çizgi rengi
- Kalınlık: Dörtgenin dış çizgisinin kalınlık seviyesi
- Dolgu: Dörtgenin içinin dolgu renkli olması isteniyorsa 1, istenmiyorsa 0 yazılır

- Dolgurenk: dolgu=1 denmişse iç dolgunun rengi verilir
- Nokta: Değer 1 yapılırsa dörtgenin köşelerinde birer nokta işareti çizilir
- Noktarenk: Köşelerde çıkan noktaların rengi belirlenir.

Örnek: Köşelerin denk geldiği bar numaraları ve fiyat seviyeleri elle girilerek bir dörtgen çizdirmek;

int SonBar=Sistem.BarSayisi;

Sistem.DortgenCiz(1, SonBar-1850, 115.90, SonBar-1180, 151.65, SonBar-200, 120.4, SonBar-570, 98.5, Color.Red, 3, 1, Color.Purple,1,Color.White);



#### - Düşük (Dusuk) Fiyat - Sistem.Dusuk(Sembol)

Bir sembolün o an ki fiyata göre çeşitli dönemlerde gördüğü **EN DÜŞÜK FİYATI** okumak için kullanılır. Fonksiyonun içine Düşük fiyatı okutulmak istenen sembol yazılır.

<u>Not:</u> iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir. Örneğin USDTYR için FX'USDTRY şeklinde yazılır. Bir sembolün PİYASA kodunun ne olduğu, o sembolü sayfanıza yazarken @ işareti yanında gösterilir.)



Kullanım şekilleri aşağıdaki gibidir.

Sistem.DusukAltiAy(Sembol); Sistem.DusukBirAy(Sembol); Sistem.DusukBirHafta(Sembol); Sistem.DusukBirYil(Sembol); Sistem.DusukBuAy(Sembol); Sistem.DusukBuHafta(Sembol); Sistem.DusukBuYil(Sembol); Sistem.DusukGun(Sembol); Sistem.DusukSeans(Sembol); Sistem.DusukUcAy(Sembol);

Örnek: GARAN hisse senedinin, bu doküman yazılırken ki son 6 ayın en düşük değerinin okuyup ekrana mesaj olarak çıkaran kod:

```
var AltiAyDusuk = Sistem.DusukAltiAy(Sembol);
Sistem.Mesaj("6 Aylık en düşük Fiyat = " + AltiAyDusuk.ToString());
```

```
var Sembol = "IMKBH'GARAN";
var AltiAyDusuk = Sistem.DusukAltiAy(Sembol);
Sistem.Mesaj("6 Aylık en düşük Fiyat = " + AltiAyDusuk.ToString());
```

6 Aylık en düşük Fiyat = 6.76

🖳 iDeal Mesaj 26.05.2020 18:19:51

#### - Ease Of Movement - Sistem.EaseOfMovement(10)

Ease Of Movement olarak bilinen indikatörünü çağırır. 1 adet parametre alır ve varsayılan paramtre olarak 10 kullanılır. **Hesaplamasında HACİM verisi kullandığından En Az DÜZEY1+ lisans gerektirir**. Aksi durumda hacim içeren barlar gecikmeli olarak kendini günceller. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.EaseOfMovement(10);
Sistem.EaseOfMovement (Veriler,10);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var EOM = Sistem. EaseOfMovement(10);
Sistem.Cizgiler[0].Deger = EOM;
```

- Ehler's Diff Coef Filter - Sistem.EhlersEhlersDiffCoefFilter()

Ehler's Filter olarak bilinen indikatörünü çağırır. Hiç parametre almaz. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.EhlersFilter();
Sistem.EhlersFilter(Veriler);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var EF = Sistem. EhlersFilter();
Sistem.Cizgiler[0].Deger = EF;
```

#### - Ehler's Filter - Sistem.EhlersFilter()

Ehler's Diff Coef Filter olarak bilinen indikatörünü çağırır. Hiç parametre almaz. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.EhlersEhlersDistCoefFilter();
Sistem.EhlersEhlersDistCoefFilter(Veriler);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var EDCF = Sistem. EhlersEhlersDistCoefFilter();
Sistem.Cizgiler[0].Deger = EDCF;
```

- Elliot Wave Oscilator (EWO) - Sistem.ElliotWaveOscilator(15,34)

Elliot Wave Oscilator (EWO) olarak bilinen indikatörünü çağırır. 2 adet parametre alır. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.ElliotWaveOscillator(12, 26);
Sistem.ElliotWaveOscillator(Veriler,12, 26);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var ewo = Sistem. ElliotWaveOscillator ();
Sistem.Cizgiler[0].Deger = ewo;
```

Emir - Sistem.Emir FonksiyonlariXXXX(5)

Robot üzerinden emir göndermek için bir emir iletmek için gerekli bütün bilgileri tanımlamak (emir adedi, emir fiyatı, emrin süresi, emrin sembolü, emrin yönü gibi) ve emri göndermek için Sistem Emir fonksiyonları kullanılır. Bu fonksiyonların tek başlarına kullanım alanları yoktur. Birkaç tanesi birlikte kullanılırlar. Emir fonksiyonlarının tam listesi ve yazım şekilleri aşağıdaki gibidir;

```
Hisse/Varant/Viop hepsi için geçerli emir fonksiyonları:
```

```
Sistem.EmirSembol = "IMKBH'GARAN";
Sistem.EmirIslem = "Alış";
Sistem.EmirMiktari = 1;
Sistem.EmirTipi = "Limit"; //veya Piyasa
Sistem.EmirFiyati = 9.50;
Sistem.EmirSuresi = "GUN";
Sistem.EmirAltHesap = "1";
Sistem.EmirHesapAdi = "123456, ABC YATIRIM";
Sistem.EmirSatisTipi = "ACIGA";
Sistem.Emir = "GUN";
```

#### Sadece VIOP için geçerli Emir Fonksiyonları

```
Sistem.EmirAksamSeansi = 1; //VERİLEN EMRİN AKŞAM SEANSINDA DA GEÇERLİ OLMASI İÇİN
Sistem.EmirSartBool = true;
Sistem.EmirSartSembol = "VIP'F_XU0300620";
Sistem.EmirSartFiyat = 132.500;
Sistem.EmirSartTipi = "ALIŞ >= Şart Fiyat";
Sistem.EmirGonder();
```

Notlar:

- Piyasa emirlerinde SÜRE = KIE olmalıdır
- Piyasa emri verilirken Emir Fiyat komutu gereksizdir.
- Emir Fiyat Tipi komutu BIST'ın nasdaq öncesi dönemden kalmadır, işlevsizdir.
- Hisselerde Açığa Satış yapılacaksa Emir Satış Tipi = "ACIGA" yazılmalıdır. Normal satış için bu komutu yazmaya gerek yoktur.
- Emri Sembol yazılırken Piyasakodu'SenetKodu şeklinde yazılmalıdır ve çift tırnak içine alınmalıdır. ("IMKBH'GARAN" veya "VIP'F\_XU0300620")
- Fiyata emir verirken Hisse için EmirTipi = Limit, VIOP için Emir Tipi = Limitli Yazılır
- Emir Gönder komutu diğer emir özellikleri ataması yapılmış olan bilgileri bir araya getirip emri aracı kuruma iletir. İDEAL Portföy penceresinde login olunmuş ve seçili durumda olan hesap ne ise emir o hesaba iletilir.
- Seçili olunmuş hesaptan başka bir hesaba emir iletilmek isteniyorsa EmirHesapAdi ve EmirAltHesap satırlarına emrin gönderilmek istendiği hesabın bilgileri ÇİFT TIRNAK işaretleri içinde yazılır.

Örnek bir hesap için bu emrin iletilmek istendiği hesap ve alt hesap bilgilerinin hangileri olduğu resimde gösterildiği gibidir;



Örnek: Her 10 saniyede bir aktiften (piyasa emri) DOHOL alış emri gönderen robot kodu

```
var Sembol = "IMKBH'DOHOL";
var Miktar = 1;
if (Sistem.ZamanKontrolSaniye(Sembol) >= 10)
{
    Sistem.ZamanKontrolGuncelle(Sembol);
    Sistem.EmirSembol = Sembol;
    Sistem.EmirIslem = "Alış";
    Sistem.EmirIslem = "Alış";
    Sistem.EmirIpi = "Piyasa";
    Sistem.EmirSuresi = "KIE";
    Sistem.EmirGonder(
}
```

Örnek 2: VIOP AKBANK Vadeli kontrata her 10 saniyede bir Şartlı emir gönderen kod

```
var Sembol = "VIP'F_AKBNK0520";
if (Sistem.ZamanKontrolSaniye(Sembol) >= 10)
{
   Sistem.ZamanKontrolGuncelle(Sembol);
   var Miktar = 1;
      Sistem.EmirSembol = Sembol;
      Sistem.EmirIslem = "Alış";
      Sistem.EmirMiktari = Miktar;
      Sistem.EmirSuresi = "GUN";
                                      // GUN, SNS, IKG
      Sistem.EmirTipi = "Limitli";
                                         // KPY, KIE, GIE, SAR
      Sistem.EmirFiyati = 5.10;
      Sistem.EmirSartSembol=Sembol;
      Sistem.EmirSartBool=true;
      Sistem.EmirSartFiyat= 4.50;
      Sistem.EmirSartTipi="ALI$ >= $art Fiyat";
      Sistem.EmirGonder();
}
```

- En Çok Tekrar - Sistem.EncokTekrar

Bir sembolün herhangi bir periyotlu grafiğinde, bir fiyatın veya bir indikatörün, en fazla sayıda tekrar etmiş olan değerini bulmaya yarayan ideal sistem fonksiyonudur. İki ayrı kullanım şekli vardır. Birinci kullanım şekli bir fiyat/indikatör listesinin <u>son x adet barı içinde en çok tekrar eden değeri</u> veren ve bu sayede bir indikatör olarak da çizilebilen kullanımdır. Tıpkı 14'lük ortalama çizdirmek gibi, hep son 14 bar içindeki en çok tekrar eden kapanış değerini alarak çiz diyerek en çok tekrar isimli bir indikatör elde edebiliriz. İkinci kullanım şekli ide, bir listenin <u>TAMAMININ içindeki en çok sayıda tekrar eden değeri</u> (tek bir değer) elde etmek amaçlı kullanımdır. Kullanım şekillerinin yazım yöntemleri aşağıdaki gibidir;

```
Sistem.EnCokTekrar(Liste, 14);
Sistem.EnCokTekrarDeger(Liste);
```

Örnek 1: Son 50 bar içerisinde en çok kez görülen kapanış değerlerini indikatör olarak hesaplayıp çizen kod.

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var IND = Sistem.EnCokTekrar(C,50);
Sistem.Cizgiler[0].Deger = IND;
```



### Örnek 2: Son 1000 bar içerisinde hangi yüksek seviyesi kaç kez görüldü bilgisini bu komutu kullanmadan elde etmek:

```
var V = Sistem.GrafikVerileri;
var H = Sistem.GrafikFiyatSec("Yuksek");
var Tekrarlar = new SortedDictionary<float, int>();
for (int i = V.Count-1000; i < V.Count; i++) // v.count-1000 den başlamak son 1000 bar demektir.
{
    float Fiyat = H[i];
    int Defa = 0;
    if (Tekrarlar.ContainsKey(Fiyat))
        Defa = Tekrarlar[Fiyat];
    Defa++;
    Tekrarlar[Fiyat] = Defa;
}
var Str = string.Join("\r\n", Tekrarlar.Select(x => x.Key.ToString("0.00") +
                                                                                      "\t
                                                                                               +
x.Value.ToString()).ToList());
Sistem.Mesaj(Str);
```



🖳 iDeal Me	saj 30.05.2020 20:50:34
7.53 1	14
7.54 2	25
7.55 2	26
7.56 1	19
7.57 4	1
7.58 7	7
7.59 5	ī
7.60 1	13
7.61 3	30
7.62 1	19
7.63 9	)
7.64 2	23
7.65 7	1
7.66 6	i i i i i i i i i i i i i i i i i i i
7.67 7	7
7.68 1	10

Örnek 3: Bir grafikte 100'lük HHV indikatörünün en çok sayıda tekrar eden değerini öğrenmek:

Not: Saatlik GARAN grafiğine uygulanmıştır.

```
var HHV = Sistem.HHV(100, "Yuksek");
var X = Sistem.EnCokTekrarDeger(HHV);
Sistem.Mesaj(X.ToString("0.00"));
```

🖳 iDeal Mesaj 30.05.2020 21:00:06

6.14

#### Envelope Up/Down/Mid - Sistem.Envelope()

Envelope isimli üst/alt/orta banddan oluşan 3 çizgisi olan indikatörü hesaplayıp çağıran ideal sistem fonksiyonudur. Biri içerisinde kullanılan Moving Average Yöntemi diğer ikisi de periyot olmak üzere 3 parametre alır. Mid (Orta), Alt (Down) ve Up (Üst) band çizgilerinin her biri için ayrı bir fonksiyon vardır. Hesaplanmasında barların OHLC değerlerinin tamamı gerektiği için Liste kullanım tipi yok, Veriler (başka bir sembolün veya periyodun bar verileri) kullanım şekli vardır. Tüm kullanım ve yazım şekilleri aşağıdaki gibidir;

```
Sistem.EnvelopeDown("Simple", 25, 5);
Sistem.EnvelopeDown(Veriler , "Simple", 25, 5);
Sistem.EnvelopeMid("Simple", 25, 5);
Sistem.EnvelopeMid(Veriler,"Simple", 25, 5);
Sistem.EnvelopeUp("Simple", 25, 5);
Sistem.EnvelopeUp(Veriler,"Simple", 25, 5);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var Alt = Sistem.EnvelopeDown("Simple", 25, 5);
var Orta = Sistem.EnvelopeMid("Simple", 25, 5);
var Ust = Sistem.EnvelopeUp("Simple", 25, 5);
Sistem.Cizgiler[0].Deger = Alt;
Sistem.Cizgiler[1].Deger = Orta;
Sistem.Cizgiler[2].Deger = Ust;
```



#### - Excel Kopyala - Sistem.ExcelKopyala

iDeal programından okutulan veya hesaplatılan bilgilerin bir excel dosyasına yazdırılması amacıyla kullanılan ideal fonksiyonudur. Fonksiyonun kullanımında CELL (Hücreler) içerisinde satır ve sütun bazında yazdırılacak verileri elde edildikten sonra (hücre dizisi) bu hücre dizisinin (cell array) bir dosya adıyla birlikte komutun içine yazılması yöntemi kullanılır. Fonksiyonun yazım şekli aşağıdaki gibidir;

Not: İşletim sisteminin kütüphanesi kullanılarak EXCEL uygulaması çağırılıp ekrana açılır ve kodda girdi olarak veriler bilgiler hücrelere yazılmış olarak excel gösterilir. Farklı Excel sürümlerinde excel açma, içine veri yazma ve kaydetme izinleri çalışmayabilir.

Sistem.ExcelKopyala(cellarray, DosyaAdı);

Örnek: Bütün hizsse senetlerinin YIL YÜKSEK bilgisini okuyan ve İDEAL klasörü altında oluşturulacak YUKSEKLER isimli bir excel dosyasına kaydeden kod.

<u>Not: Formülü yazıp FORMÜL TEST butonuna basmak, kod içinde yazılı işlemlerin bir defalığına</u> yapılması için yeterlidir. Formülü robot olarak çalıştırmak veya sistem olarak grafiğe uygulamak <u>gerekmez.</u>

```
var Liste = Sistem.YuzeyselListeGetir("IMKBH'");
var ExcelFileName = "\\ideal\\YUKSEKLER.Xlsx";
var cellarray = new object[Liste.Count + 1, 10];
for (int i = 0; i < Liste.Count; i++)
{
    cellarray[i + 1, 0] = Liste[i].Symbol;
    cellarray[i + 1, 1] = Liste[i].HighYear.ToString();
}
Sistem.ExcelKopyala(cellarray, ExcelFileName);
```

#### Kodun açıklamasını satır satır şöyle yapabiliriz;

**<u>1.Satır:</u>** Bu satırda piyasası IMKBH olan (tüm hisseler, varantlar) tüm semboller için Yüzeysel Veri Listesi çağrılır

**<u>2.Satır:</u>** Açılacak ve kaydedilecek excel dosyasının diskteki yeri ve isminden oluşan bir DOSYA ADI (File Name) tanımlanır.

**<u>3.Satır:</u>** İlk satırda çağrılan sembol verilerini içeren Listenin eleman sayısından 1 fazla sayıda satırı, 10 sütunu olan bir excel objesi (nesnesi) tanımlanır. (cell/hücre dizisi) (sütun sayısı rastgele 10)

**4.5.6.7.8.Satırlar:** Bu satırlarda bir FOR DÖNGÜSÜ vardır. İlk satırda çağrılan sembol verilerini içeren Listenin elemanlarının HER BİRİ İÇİN DÖN ve her biri için, onun liste içindeki satır numarasına denk gelen numaranın 1 fazla numaralı excel satırına

\* SIFIR nolu sütuna (ilk sütun) sembolü yaz

\* 1BİR nolu sütuna (ikinci sütun) YÜZEYEL LİSTE isimli veri listesinini Yıl Yüksek bilgisini (HighYear) yaz

**<u>9.Satır:</u>** *ExcelKopyala fonksiyonunun çalışması için gereken input (giriş) bilgilerini fonksiyona ver (hücre dizisini ve dosya adını)* 

#### - Excel Oku - Sistem.ExcelOku

Bilgisayarınızda var olan bir Excel dosyasını okumak, söz konusu Excel dosyası içerisinden okutulacak verilere göre işlemler yapmak amacıyla kullanılan ideal fonksiyonudur. Fonksiyonun yazım şekli aşağıdaki gibidir;

```
Sistem.ExcelOku(Kitap.Xlsx);
```

Örnek: Bazı hisse senetlerini, kendi belirleyeceğimiz fiyat seviyelerine düşünce alan, yine bizim excel sütunlarına yazacağımız KAR AL veya STOP fiyatlarına gelince SATAN ve pozisyonu kapatan, her bir hisse için kaç lot işlem yapacağımızı da EXCEL üzerinden okuyarak çalışan bir ideal robot kod örneği:

```
string FileName = "C:\\ideal\\HisseRobot.Xlsx";
bool DevamEt = true;
if (DateTime.Now.DayOfWeek == DayOfWeek.Saturday) DevamEt = false;
if (DateTime.Now.DayOfWeek == DayOfWeek.Sunday) DevamEt = false;
if (DateTime.Now.ToString("HHmm").CompareTo("1000") <= 0) DevamEt = false;
if (DateTime.Now.ToString("HHmm").CompareTo("1800") >= 0) DevamEt = false;
if
              (DateTime.Now.ToString("HHmm").CompareTo("1300")
                                                                                              0
                                                                                >=
DateTime.Now.ToString("HHmm").CompareTo("1400") <= 0) DevamEt = false;</pre>
if (System.IO.File.Exists(FileName) == false) DevamEt = false;
if (DevamEt)
ł
    var ExcelArray = Sistem.NesneGetir(FileName + ";" + DateTime.Now.ToString("yyyyMMdd"));
    if (ExcelArray == null)
    {
         ExcelArray = Sistem.ExcelOku(FileName);
         Sistem.NesneKaydet(FileName + ";" + DateTime.Now.ToString("yyyyMMdd"), ExcelArray);
    }
```

SEZAİ KILIÇ

&&

```
//Excel dosyasını 10 dk'da bir tekrar oku.
    if (Sistem.ZamanKontrolDakika(FileName + ";" + DateTime.Now.ToString("yyyyMMdd")) >= 5)
    {
        Sistem.ZamanKontrolGuncelle(FileName + ";" + DateTime.Now.ToString("yyyyMMdd"));
        ExcelArray = Sistem.ExcelOku(FileName);
        Sistem.NesneKaydet(FileName + ";" + DateTime.Now.ToString("yyyyMMdd"), ExcelArray);
    }
    int SatirSayisi = ExcelArray.GetLength(0); //satirlarin sayisini bul
    for (int i = 2; i <= SatirSayisi; i++)</pre>
    {
        var Sembol = ExcelArray[i, 1].ToString();
        var AlisFiyat = (decimal)ExcelArray[i, 2];
        var HedefFiyat = (decimal)ExcelArray[i, 3];
        var StopFiyat = (decimal)ExcelArray[i, 4];
       double IslemFiyat = 0;
        DateTime IslemTarih;
        var Rezerv = "";
        var Pozisyon = Sistem.PozisyonKontrolOku(Anahtar, out IslemFiyat, out IslemTarih, out
Rezerv);
        var EmirSembol = "IMKBH'" + Sembol;
        var basicitem = Sistem.YuzeyselVeriOku(EmirSembol);
        var sonfiyat = (decimal)basicitem.LastPrice;
        var bidfiyat = (decimal)basicitem.BidPriceDec;
        var askfiyat = (decimal)basicitem.AskPriceDec;
        if (sonfiyat == 0) continue;
        if (bidfiyat == 0) continue;
        if (askfiyat == 0) continue;
        var Islem = "";
        var Miktar = 0.0;
        if (sonfiyat <= AlisFiyat && Pozisyon == 0 && Rezerv == "") // AL
        {
           Rezerv = "AL";
           Miktar = Lot;
           IslemFiyat = Sistem.SonFiyat(EmirSembol);
        }
        else if (Pozisyon > 0 && askfiyat >= HedefFiyat && Rezerv == "AL") // KARLA KAPAT
        {
           Rezerv = "KAR AL";
           Miktar = -Lot;
        }
        else if (Pozisyon > 0 && bidfiyat < StopFiyat && Rezerv == "AL") // STOP
        {
           Rezerv = "STOP";
           Miktar = -Lot;
        }
        if (Miktar > 0) Islem = "ALIS";
        if (Miktar < 0) Islem = "SATIS";</pre>
        if (Islem != "")
        {
           Sistem.PozisyonKontrolGuncelle(Anahtar, Miktar + Pozisyon, IslemFiyat, Rezerv);
           Sistem.EmirSembol = EmirSembol:
           Sistem.EmirIslem = Islem;
           Sistem.EmirSuresi = "KIE"
           Sistem.EmirTipi = "Piyasa";
           Sistem.EmirMiktari = Math.Abs(Miktar);
           Sistem.EmirGonder();
       }
  }
}
```

<u>Söz konusu örnek robot tarafından okutulan Excel dosyası içi aşağıdaki fotoda gösterildiği</u> gibidir;

	А	В	С	D	E	
1	SENET	AL	KAR AL	STOP	Miktar	
2	GSRAY	5.30	5.40	5.00	100	
3	FENER	3,56	3.64	3,48	50	
4	BJKAS	2.31	2,45	2.21	25	
5						
6						

#### Yukarıdaki Örnek Robot kodu için ek bilgiler;

\*İlk satırda bilgilerimizi yazdığımız Excel dosyasının adı ve bilgisayarımızda bulunduğu yer tanımlanmıştır.

\*Sonraki satırda, aşağıda belirtilen zaman aralıklarında (hafta sonları, sabah 10'dan önce ve akşam 18'den sonra) robotun çalışmaması ayarları yapılmıştır.

Diğer Satırlar idealin Sistem ve Robot kütüphanesinde henüz bu sayfaya kadar anlatılmamış başka bazı fonksiyonları, C# dilinin bazı komutları kullanılmış olsa da, bu amaçla söz konusu örneği kullanmak isteyenlerin değiştirmeden kullanabileceği kodlardır.

#### - Fark - Sistem.Fark(Sembol)

Bir sembolün o an ki fiyata göre çeşitli dönemlere göre DEĞİŞİMİNİ (Puan veya Para olarak nominal) farkını okumak için kullanılır. Fonksiyonun içine FARKI/DEĞİŞİMİ okutulmak istenen sembol yazılır.

Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.FarkAltiAy(Sembol);
Sistem.FarkBirAy(Sembol);
Sistem.FarkBirHafta(Sembol);
Sistem.FarkBirYil(Sembol);
Sistem.FarkBuAy(Sembol);
Sistem.FarkBuHafta(Sembol);
Sistem.FarkGun(Sembol);
Sistem.FarkGun(Sembol);
Sistem.FarkSeans(Sembol);
Sistem.FarkUcAy(Sembol);
```

Örnek: XU100 endeksinin, son 1 haftalık FARK/DEĞİŞİM değerini okuyup ekrana mesaj olarak çıkaran kod:

```
var Degisim = Sistem.FarkBirHaftaAltiAy("IMKBX'XU100");
Sistem.Mesaj("1 Haftalık değişim = " + Degisim.ToString());
```



#### - Fibonacci Up/Down/Mid - Sistem.Fibonacci()

Fibonacci bands olarak bilinen üst/alt/orta çizgilerden oluşan indikatörü hesaplayıp çağıran ideal sistem fonksiyonudur. Varsayılan değeri 1 olan 1 adet parametre alır. Mid (Orta), Alt (Down) ve Up (Üst) band çizgilerinin her biri için ayrı bir fonksiyon vardır. Hesaplanmasında barların OHLC değerlerinin tamamı gerektiği için Liste kullanım tipi yok, Veriler (başka bir sembolün veya periyodun bar verileri) kullanım şekli vardır. Tüm kullanım ve yazım şekilleri aşağıdaki gibidir;

```
Sistem.FibonacciDown(1);
Sistem.FibonacciDown(Veriler,1);
Sistem.FibonacciMid(1);
Sistem.FibonacciMid(Veriler,1);
Sistem.FibonacciUp(1);
Sistem.FibonacciUp(Veriler,1);
```

#### Örnek: İndikatörün 3 bandını da çağırıp çizdirmek

```
var Alt = Sistem.FibonacciDown(1);
var Orta = Sistem.FibonacciMid(1);
var Ust = Sistem.FibonacciUp(1);
```

```
Sistem.Cizgiler[0].Deger = Alt;
Sistem.Cizgiler[1].Deger = Orta;
Sistem.Cizgiler[2].Deger = Ust;
```



#### - Forecast Osc - Sistem.ForecastOsc(5)

Forecast Oscilator isimli indikatörü hesaplayıp çağıran ideal sistem fonksiyonudur. 1 adet parametre alır ve varsayılan değeri 5'tir. Hesaplanmasında barların OHLC değerlerinin tamamı gerektiği için Liste kullanım tipi yok, Veriler (başka bir sembolün veya periyodun bar verileri) kullanım şekli vardır. Grafik barları üzerine değil, alt bölgedeki panellere çizilir. Kullanım ve yazım şekilleri aşağıdaki gibidir;

```
Sistem.ForecastOsc(5);
Sistem.ForecastOsc(Veriler,);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var X = Sistem. ForecastOsc(5);
Sistem.Cizgiler[0].Deger = X; //Panel no 2 yapılmalıdır
```

- FRAMA - Sistem.FRAMA()

FRAMA isimli indikatörü hesaplayıp çağıran ideal sistem fonksiyonudur. Hiç Parametre almaz. Grafik barları üzerine çizilerek FİYAT ile korelasyonu takip edilir. Kullanım ve yazım şekilleri aşağıdaki gibidir;

Sistem.FRAMA();
Sistem.FRAMA(Veriler);

Örnek: İndikatörü çağırıp çizdirmek

```
var X = Sistem. FRAMA();
Sistem.Cizgiler[0].Deger = X; //Panel no 1 yapılmalıdır
```

```
- FX Sniper - Sistem.FxSniper(3,2)
```

FX Sniper isimli indikatörünü çağırır. 2 adet parametre (periyot) alır ve varsayılan değer olarak 3 ve 2kullanılır. Aşağıdaki gibi 3 kullanım şekli vardır;

```
Sistem.FxSniper(3,2);
Sistem.FxSniper(Liste,3,2);
Sistem.FxSniper(Veriler,3,2);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var FX = Sistem.FxSniper(3,2);
Sistem.Cizgiler[0].Deger = FX;
```

- Getiri Hesapla (GetiriKZ) - Sistem.GetiriHesapla(Tarih, Komisyon)

**Sistem.GetiriHesapla** fonksiyonu, bir sistem formülüne yazıldığı zaman, sisteminizin stratejisi, bu fonksiyona girilen başlangıç tarihinden itibaren son bara kadar hesaplanır ve biz kayıt

listesinde tutulur. Hesaplama yapılırken, komisyon veya kayma maliyetlerini de girerseniz, o maliyetleri çıkarıp getiri hesabı yapar.

Stratejinizi yazıp grafiğe uyguladınız, al/sat noktalarını görüyorsunuz, ama hemen altında da getiri/KZ grafiği eklemek istiyorsanız bu fonksiyonla getiriyi hesaplatabilirsiniz. Geriye kalan, sonucu bir çizgiye atamak.

Fonksiyonu, başlangıç tarihi ve isteniyorsa kayma/komisyon maliyeti için işlem başına düşülmesi istene puan/fiyat bilgisi girerek, aşağıdaki şekilde kullanabilirsiniz:

#### Sistem.GetiriHesapla("01/01/2010", 0.00);

Yapmanız gereken şey getiriyi (istediğiniz bir tarihten başlayarak) hesaplayan fonksiyonu sisteminizin en altına yazmak ve sonucu bir çizgiye atamaktır. (Yani kodunuza aşağıdaki gibi iki satırı eklemek)

Bu iki satır aşağıdaki gibidir:

```
Sistem.GetiriHesapla("10/10/2010",0.000);
Sistem.Cizgiler[0].Deger = Sistem.GetiriKZ; //Panel No= 2
```

Getiri Hesapla fonksiyonu, sistemin işleme girdiği barları, işleme girdiği fiyatları, pozisyonun yönünü ve miktarını bulur ve çeşitli listelere doldurur. Tek başına bu fonksiyonu koda yazmak yeterli değildir. Bu fonksiyon ile elde edilen/hesaplanan verilerin her birine aşağıda tam listesi verilen birer alt fonksiyon ile ulaşılır. Bu fonksiyonların bazıları liste verir (grafiğe çizgi olarak çizdirilenler). Bir kısmı ise sayı/oran/değer gibi bir tek sonuç (performans analiz raporundaki önemli bilgiler) olarak elde edilir. (formülde kullanmak, ekrana yazdırmak vs. amaçlı)

#### **GRAFIĞE ÇİZDİRİLEBİLEN GETİRİ HESAPLA LİSTELERİ:**

```
Sistem.GetiriKZGunSonu;
Sistem.GetiriKZGunBasi;
Sistem.GetiriKZAySonu;
Sistem.GetiriKZAyNet;
Sistem.GetiriKZYilNet;
Sistem.GetiriKZGun;
Sistem.GetiriKZGun;
Sistem.GetiriKZAy;
Sistem.GetiriKZYil;
Sistem.GetiriPozisyon;
```

#### PUAN/ORAN/SAYI VERİLERİ SUNAN GETİRİ HESAPLA SONUÇLARI:

```
Sistem.GetiriIslemSayisiPoz;
Sistem.GetiriIslemSayisiGun;
Sistem.GetiriIslemSayisiAy;
Sistem.GetiriIslemSayisiYil;
Sistem.GetiriMutluGun;
Sistem.GetiriKarIslem;
Sistem.GetiriZararIslem;
Sistem.GetiriZararIslem;
Sistem.GetiriToplamIslem;
Sistem.GetiriKarMiktar;
Sistem.GetiriZararMiktar;
Sistem.ProfitFactor;
```



Sistem.GetiriNetKar; Sistem.GetiriBuAy; Sistem.GetiriBirAy;

ÖRNEK1: TOMA stratejisi kullanılan bit sistemde, GETİRİ HESAPLA fonksiyon ile dönen listeler kullanılmış ve aşağıdaki örnekte ekrana çizdirilmiştir.

```
var TOMA = Sistem.TOMA(3, 2.48);
var Veriler = Sistem.GrafikFiyatSec("Kapanis");
var EMA = Sistem.MA(Veriler, "Exp", 3);
```

```
Sistem.KesismeTara(EMA, TOMA);
var Renk2 = Sistem.Renk(60,255,0,100);
var Renk1 = Sistem.Renk(60,0,255,0);
```

Sistem.GetiriHesapla("01/01/2010", 0.00);

Sistem.Cizgiler[0].Deger	=	Sistem.GetiriKZ;
Sistem.Cizgiler[1].Deger	=	Sistem.GetiriKZPoz;
Sistem.Cizgiler[2].Deger	=	Sistem.GetiriKZGun;
Sistem.Cizgiler[3].Deger	=	Sistem.GetiriKZGun;
Sistem.Cizgiler[4].Deger	=	Sistem.GetiriKZYil;
Sistem.Cizgiler[5].Deger	=	Sistem.GetiriMiktar;
Sistem.Cizgiler[6].Deger	=	<pre>Sistem.GetiriPozisyon;</pre>

No	Açıklama	Aktif	Panel	Renk	Kalinlik	Stil
0	KarZarar	$\checkmark$	2	•	2	1 : Düz
1	İşlem Bazında KZ	$\checkmark$	3	٠	1	1 : Düz
2	Gün Bazında KZ	$\checkmark$	4	٠	1	1 : Düz
3	Ay Bazında KZ		5	٠	1	1 : Düz
4	Yıl Bazında KZ	$\checkmark$	6		1	1 : Düz
5	İşlem Miktarı	$\checkmark$	7	•	1	1 : Düz
6	Anlık Pozisyon		8	•	1	1 : Düz



ÖRNEK2: SUM fonksiyonu kullanılarak oluşturulan bir strateji kullanan sistemde, her bir işlem bazında KAR/ZARARI çizdiren, Performans Analizindeki bazı önemli sonuçları da grafik üzerine yazdıran örnek (İLK 2 ÇİZGİ AKTİF EDİLİR, PANEL NUMARALARI 2 YAPILIR)

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var A = Sistem.Sum(C,10);
var B = Sistem.Liste(0);
for (int i=3; i<Sistem.BarSayisi; i++)</pre>
  B[i] = A[i]+A[i-1]-A[i-2]-A[i-3];
var SonYon ="";
for (int i=1; i<Sistem.BarSayisi; i++)</pre>
{
   if(B[i] > 3.60f && SonYon!="A")
   {
      Sistem.Yon[i] = "A";
      SonYon = "A";
   }
   else if(B[i] < -3.60f && SonYon!="S")</pre>
   {
      Sistem.Yon[i] = "S";
      SonYon = "S";
   }
}
Sistem.GetiriHesapla("01/01/2010",0.000);
var Renk1 = Sistem.Renk(150, 255, 255, 0);
var Renk2 = Sistem.Renk(255, 0, 255, 0);
var Renk3 = Sistem.Renk(255, 255, 0, 80);
```

Sistem.ZeminYazisiEkle("Toplam İşlem Sayısı = " + " " +Sistem.GetiriToplamIslem.ToString("0"), 1, 170, 20, Renk1, "Tahoma", 10); Sistem.ZeminYazisiEkle("Kazandıran İşlem Sayısı = " + " " + Sistem.GetiriKarIslem.ToString("0"), 1, 170, 35, Renk1, "Tahoma", 10); н ... Sistem.ZeminYazisiEkle("Kaybettiren İşlem Sayısı = Sistem.GetiriZararIslem.ToString("0"),1, 170, 50, Renk1, "Tahoma", 10); Sistem.ZeminYazisiEkle("Karlı İşlem Miktarı = " + " " + Sistem.GetiriKarMiktar.ToString("0.000"), 1, 220, 85, Renk1, "Tahoma", 10); ... ... Sistem.ZeminYazisiEkle("Zararlı İşlem Miktarı + = Sistem.GetiriZararMiktar.ToString("0.000"),1, 220, 100, Renk1, "Tahoma", 10); Sistem.ZeminYazisiEkle("Karlı İşlem Oranı = " + "% " + Sistem.GetiriKarIslemOran.ToString("0.00"), 1, 220, 115, Renk1, "Tahoma", 10); Sistem.ZeminYazisiEkle("Profit Factor = " + " " + Sistem.ProfitFactor.ToString("0.00"),1, 440, 40, Renk3, "Tahoma", 12); Sistem.ZeminYazisiEkle("Net Kar = " + " " + Sistem.GetiriNetKar.ToString("0.000"),1, 440, 20, Renk2, "Tahoma", 15);

Sistem.Cizgiler[0].Deger = Sistem.GetiriKZPoz; Sistem.Cizgiler[1].Deger = Sistem.Liste(0);

var Renk10 = Sistem.Renk(60,255,0,0); var Renk20 = Sistem.Renk(60,0,255,0); Sistem.DolguEkle(1,0,Renk10,Renk20);



ÖRNEK-3: Pozisyon bazında kar/zarar görmek için Sistemin altına eklenmiş satırlar: (Sistem bir pozisyona girdiği zaman, kapatılmış olan bir önceki pozisyonda ne kadar kar yada zarar edildiği gösterilir)

Sistem.GetiriHesapla("10/10/2010",0.000); Sistem.Cizgiler[0].Deger = Sistem.GetiriKZPoz;

Sonuc:





ÖRNEK-4: Sisteminiz işlem yaptıkça, pozisyonunuz ne (kaç lot) olmuş, her sinyalde kaç adet/lot işlem yapılmış ve getiriniz nasıl seyretmiş bilgilerini görmek için Sistemin altına eklenmiş satırlar:

```
Sistem.GetiriHesapla("10/10/2010",0.000);
Sistem.Cizgiler[0].Deger = Sistem.GetiriKZ;
Sistem.Cizgiler[1].Deger = Sistem.GetiriPozisyon;
Sistem.Cizgiler[2].Deger = Sistem.GetiriMiktar;
```

Sonuç:





ÖRNEK-5: Sisteminiz pozisyon başına kar/zararı alt kısma çizmek, grafik zemini üzerine de performans analizi sonuçlarından elde edilen verileri yazmak için aşağıdaki kodları ekleyebilirsiniz. NOT: Çizgi renkleri, dolguda kullanılan renkler ve yazıların grafik üzerindeki x/y koordinatları sizin tercihinize ve ayarlarınıza göre değişen bilgilerdir.

```
Sistem.GetiriHesapla("01/01/2010",0.000);
Sistem.GetiriMaxDDHesapla("01/10/2015","01/01/2020");
var MaxDD = Sistem.GetiriMaxDD.ToString();
var MaxDDTarihi = Sistem.GetiriMaxDDTarih.ToString("dd/MM/yyyy");
                                                                    =" + MaxDDTarihi
Sistem.ZeminYazisiEkle("Max DD = "+ MaxDD
                                                      + "(Tarih
                                                                                                   ")"
,1,80,350,Color.Gold,"Tahoma",20);
Sistem.ZeminYazisiEkle("Toplam İşlem Sayısı = " + " " +Sistem.GetiriToplamIslem.ToString("0"), 1,
170, 20, Renk1, "Tahoma", 10);
Sistem.ZeminYazisiEkle("Kazandıran İşlem Sayısı = " + " " + Sistem.GetiriKarIslem.ToString("0"),
1, 170, 35, Renk1, "Tahoma", 10);
                                                                                       ...
                                                                                              ...
Sistem.ZeminYazisiEkle("Kaybettiren
                                          İşlem
                                                    Sayısı
Sistem.GetiriZararIslem.ToString("0"),1, 170, 50, Renk1, "Tahoma", 10);
Sistem.ZeminYazisiEkle("Karlı İşlem Miktarı = " + " " + Sistem.GetiriKarMiktar.ToString("0.000"),
1, 220, 85, Renk1, "Tahoma", 10);
Sistem.ZeminYazisiEkle("Zararlı
                                                                       ....
                                                                                             н
                                      İşlem
                                                  Miktarı
                                                                              +
Sistem.GetiriZararMiktar.ToString("0.000"),1, 220, 100, Renk1, "Tahoma", 10);
Sistem.ZeminYazisiEkle("Karlı İşlem Oranı = " + "% " + Sistem.GetiriKarIslemOran.ToString("0.00"),
1, 220, 115, Renk1, "Tahoma", 10);
Sistem.ZeminYazisiEkle("Profit Factor = " + " " + Sistem.ProfitFactor.ToString("0.00"),1, 440, 40,
Renk3, "Tahoma", 12);
Sistem.ZeminYazisiEkle("Net Kar = " + " " + Sistem.GetiriNetKar.ToString("0.000"),1, 440, 20,
Renk2, "Tahoma", 15);
Sistem.Cizgiler[0].Deger = Sistem.GetiriKZPoz;
Sistem.Cizgiler[1].Deger = Sistem.Liste(0);
```

Sonuç:




**NOT1: Profit Factor:** Getiri hesaplamasında, en önem verilen sonuçlardan biri de PROFIT FACTOR bilgisidir. Profit Factor bir sistemde kazandıran sinyallerin kazandığı getirinin, kaybettiren sinyallerin kaybettirdiği para/puan değerine oranıdır. Sistem kütüphanesinde Getiri Hesapla komutu kullanıldığı zaman sisin stratejinizin tüm grafik zamanı için Profit Factor değerinin ne olduğu bilgisine yukarıdaki SAYI/ORAN dönen sonuç listesindeki Sistem.ProfiFactor komutuyla ulaşılabilir. Bununla birlikte Sistem.ProfitFactor isimli toplam 4 fonksiyon bulunmaktadır. Bu fonksiyonların kullanım/yazım şekilleri aşağıdaki gibidir;

Sistem.ProfitFactor; //tek bir oran olarak tüm sistemin Profit Factorü Sistem.ProfitFactorBar(BarSayisi, Kayma); //Girilen bar sayısı kadar öncesine ait PF Sistem.ProfitFactorIslem(IslemSayisi, Kayma); //Belli işlem sayısı için PF Sistem.ProfitFactorList; //Her bir barda yeniden hesaplanan PF

NOT2: MaxDD Hesaplama: Getiri hesaplamasında, yine çok önem verilen sonuçlardan biri de MAXDD bilgisidir. MaxDD, bir sistemin kullanılması durumunda, geçmişten günümüze portföyde en büyük parasal erime miktarıdr. Bu sistemi kullansaydık porftöyümüzde en büyük erime hangi tarihte olurdu ve bu erime kaç puan/lira olurdu bilgisi için MacDD verisine bakılır. MaxDD bilgilerine erişim için, sistemin içine (en altına) Sistem.GetiriHesapla Satırından sonra Sistem.MaxDDHesapla satırını da eklemek gerekir. Sonuç değerlerine ancak o şekilde ulaşmak mümkündür. MaxDD için sunulan fonksiyonların kullanım/yazım şekilleri aşağıdaki gibidir;

Sistem.GetiriMaxDDHesapla(Tarih1, Tarih2); Sistem.GetiriMaxDDTarih; Sistem.GetiriMaxDD;

- Görüntü Kaydet - Sistem.GoruntuKaydet(DosyaAdı)

iDeal kullanıcıları, diledikleri an bir koşul veya bir zaman geldiğinde bilgisayar ekranlarının fotoğrafını (ekran yakalama / screen capture) çekebilir ve bilgisayara kaydedebilirler ve hatta kaydettikleri bu fotoğrafı kendilerine (veya başkalarına) mail olarak iletebilirler. (Kılavuzun Mail Gönder fonksiyon anlatımında bu ikinci seçeneğe ilişkin örnek yer almaktadır) Fonksiyonun kullanım/yazım şekli aşağıdaki gibidir;

```
Sistem.GoruntuKaydet("C:\\test.png");
```

Dosya Adı değişkeni, dosyanın kaydedileceği yer ile birlikte yazılmalıdır ve tüm isim (yer+ad) çift tırnak içerisinden yazılır.

<u>Örnek: Sisteminizde AL veya SAT koşulu oluştuğu zaman ekranın fotoğrafını çekip kaydeden</u> <u>kod:</u>

```
var Veriler = Sistem.GrafikVerileri;
var C= Sistem.GrafikFiyatSec("Kapanis");
var MA1 = Sistem.MA(C, "Simple", 5);
var MA2 = Sistem.MA(C, "Simple", 22);
Sistem.Cizgiler[0].Deger = MA1;
Sistem.Cizgiler[1].Deger = MA2;
var dosyaadi = "C:\\IdealResim_" + DateTime.Now.ToString("yyyyMMddHHmm")+".png";
var SonYon = "";
for (int i = 1; i<Sistem.BarSayisi; i++)</pre>
{
  if (MA1[i-1] < MA2[i-1] && MA1[i] > MA2[i] && SonYon != "A") // 1.ortalama 2.ortalamayı yukarı
keserse
  {
     SonYon = "A";
     Sistem.Yon[i] = "A";
     Sistem.GoruntuKaydet("C:\\test.png");(dosyaadi);
   }
   if (MA1[i-1] > MA2[i-1] && MA1[i] < MA2[i] && SonYon != "S") // 1.ortalama 2.ortalamayı aşağı
keserse
   {
     SonYon = "S";
     Sistem.Yon[i] = "S";
     Sistem.GoruntuKaydet(dosyaadi);
   }
}
```

- Grafik Verileri – Grafik Fiyat Seç – Grafik Fiyat Oku

Bir Sembolün GRAFİK verilerini okumak için kullanılan ideal sistem fonksiyonlarıdır. Fonksiyonların kullanım/yazım şekilleri aşağıdaki gibidir;

```
Sistem.GrafikVerileri();
Sistem.GrafikVerileriniOku(Sembol,Periyot);
Sistem.GrafikFiyatSec("Deger");
Sistem.GrafikFiyatOku(GrafikVerileri, "Değer");
Sistem.GrafikFiyatOku("IMKBH'YKBNK","G", "Kapanis");
```

• GrafikVerileri:

# ideal

Grafiğin üzerine uygulanacak bir formül (bir indikatör veya bir sistem) yazılırken kullanılır. Hangi sembolün hangi periyotlu grafiğinin verilerinin okunacağını, grafiğe uygulanınca kendisi anlar. Dönüş değeri satır ve sütunlardan oluşan (record) bir listedir. Listenin her bir satırı bir barı, her bir sütunu da o barın bir değerini (Open, High,Low,Close, Date,Time,Volume,Size vs) içerir. Bu fonksiyonla okutulan Veriler tablosundan bir veri tipi Veriler.Deger şeklinde çekilir.

### ÖRNEK:

#### var V = Sistem.GrafikVerileri;

GrafikVerileri (veya GrafikVerileriniOku) fonksiyonu ile çekilen veri listesinde aşağıdaki Rercord'lar(Kayıtlar) yer alır. Her bir record(kayıt), **VeriListesiAdı.Kayıt** şeklinde bir komutla çekilebilir:

Veriler[Barno].Open veya Veriler[BarNo].Date gibi..

- Open
- High
- Close
- Low
- Date (Tarih&Saat fonksiyonunu detaylı inceleyiniz. C# Tarih metodlarının tamamını bu RECORD(KAYIT) için kullanabilirsiniz. (Aşağıda verilen örnekte, bir barın hem tarihi okunmuş hem de o tarih bilgisi içinden, barın haftanın hangi gününe ait olduğu bilgisi de çekilmiştir.)
- Vol (Bar'a ait işlem hacmi)
- Size (Bar'a ait işlem adedi)

ÖRNEK: Bir VOP kontratının (Ekim 2015 Vadeli) 5 dakikalık grafiğini VERİLER isimli bir listeye okutup, Bu Grafik listesinin 2500 numaralı (en eski bar NUMARASI SIFIRDIR) BAR'ı için yukarıdaki kayıtları (record'ları) okuyup DEBUG panelinde görelim;

```
var Sembol = "VIP'F_XU030101550";
var Veriler = Sistem.GrafikVerileriniOku(Sembol, "5");
Sistem.Debug("1200 Nolu Barın KAPANIS FIYATI = " + Veriler[2500].Close.ToString());
Sistem.Debug("1200 Nolu Barın YÜKSEK FIYATI = " + Veriler[2500].High.ToString());
Sistem.Debug("120 Nolu Barın DÜŞÜK FIYATI = " + Veriler[2500].Low.ToString());
Sistem.Debug("120 Nolu Barın Günü = " + Veriler[2500].Date.DayOfWeek.ToString());
Sistem.Debug("120 Nolu Barın TARİH VE SAATİ = " + Veriler[2500].Date.ToString());
Sistem.Debug("120 Nolu Barın İŞLEM HACMİ = " + Veriler[2500].Vol.ToString());
Sistem.Debug("120 Nolu Barın İŞLEM ADEDİ = " + Veriler[2500].Size.ToString());
```

*	Kaydet	Yardım	120 Nolu Barın İŞLEM ADEDİ = 4 120 Nolu Barın İŞLEM HACMİ = 38885 120 Nolu Barın TARİH VE SAATİ = 14.08.2015 16:50:00
	Şifrele	Formul Test	
	Sil	Debug	120 Nolu Barın Günü = Friday 120 Nolu Barın DÜŞÜK FIYATI = 97.175
	Yazı Tipi	Ondalık 4	1200 Nolu Barin YÜKSEK FIYATI = 97.225 1200 Nolu Barin KARANIS FIYATI = 97.175
	🔲 Indikator Penceresinde Göster		

### • GrafikVerileriniOku(Sembol, Periyot):

Yazılan formül bir grafiğin üzerine uygulanmayacak ise (Robot veya başka amaçlı bir formül yazılıyorsa) Grafik Verilerini okutmak için, hangi sembolün hangi periyotlu grafiğini okumak istediğimizi belirtmemiz gerekir. Sembolü belirtirken PIYASA'KOD şeklinde kullanmak gerekir.

#### ÖRNEK:

var V1 = Sistem.GrafikVerileriniOku("FX'EURUSD", "5");

var V2 = Sistem.GrafikVerileriniOku("IMKBH'SAHOL", "G");

var V3 = Sistem.GrafikVerileriniOku("VIP'VIP-X030", "60");

Dönüş Değeri, üstteki fonksiyonla aynıdır. İlgili sembolün belirtilen periyoduna ait grafiğinin tüm barları ve her bir bara ait saklanan tüm verileri bir Record olarak döner.

### • GrafikFiyatSec("Deger"):

Grafiğin üzerine uygulanacak bir formül (bir indikatör veya bir sistem) yazılırken kullanılır. Hangi sembolün hangi periyotlu grafiğinin verilerinin okunacağını, grafiğe uygulanınca kendisi anlar. Dönüş değeri seçilen grafik veri tipini (Değer) içeren bir LİSTE dir. (Örneğin Her barın Kapanış değerleri).

Bu fonksiyon ile grafiğin barlarının aşağıdaki bilgileri (parantez içine çift tırnak içinde bunlardan biri yazılarak) okutulabilir:

- Acilis
- Kapanis
- Dusuk
- Yuksek
- Hacim
- Size
- OHLC/4
- Tipik
- Ortalama
- Orta Nokta

### ÖRNEK:

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var Vol = Sistem.GrafikFiyatSec("Hacim");
var H = Sistem.GrafikFiyatSec("Yuksek");
```

• GrafikFiyatOku(GrafikVerileri, "Deger"):

BAR değerlerinin tamamını okuyan fonksiyonlardan biriyle (en başta anlatılan iki fonksiyondan biri) bir Veri (record) okutulmuş, bu Veri listesinden de barlara ait bir başka veri tipi elde edilmek isteniyorsa bu fonksiyon kullanılır. Bir üstte kullanılan fonksiyonun mantık ve format olarak aynısıdır. Tek farkı (algo veya robotlarda kullanıldığından) öncesinde hangi sembolün hangi periyoduna ait veriler kullanılacaksa, o veri listesinin belirtilmesidir.



ÖRNEK:

var Barlar = Sistem.GrafikVerileri("IMKBH'GARAN", "G"); //Garanti Bankası Hissesinin Günlük grafiklerinin tüm barları okutuldu var Volume = Sistem.GrafikFiyatOku(Barlar, "Hacim"); //Okutulan bu veri listesinden, her barın hacim değeri okutuldu.

Volume artık, her bir bara ait hacim değerini (sayısal değer olarak) içeren bir listedir. İstenirse grafik üzerinde çizgi olarak da çizdirilebilir.

### • GrafikFiyatOku("IMKBH'YKBNK","G","Kapanis");

Bir sembolün, BAR verilerini okutmadan, direkt olarak barlara ait bir veri tipini okutmak için kullanılır. Parantez içindeki parametrelere sırasıyla ve çift tırnaklar içinde sembol, periyot ve veri tipi girilir. Dönüş değeri bir sayı listesidir.

- Grafik Güncelle – Sistem.GrafikGuncelle(Sembol)

Grafik Güncelle fonksiyonunu kullanıcı tarafından yaratılmış/tanımlanmış sembollerin formüllerinin ürettiği SONFİYAT bilgisini kaydedip, o sembole ait bir grafik verisi oluşmasını / birikmesini sağlamak amacıyla kullanılır.

Fonksiyonun yazım/kullanım şekli aşağıdaki gibidir;

Sistem.GrafikGuncelle(Sembol);

Bu fonksiyon Sistem.SembolTanimla ve Sistem.YuzeyselGuncelle fonksiyonları ile birlikte kullanılır. Söz konusu iki fonksiyonu klavuzun ilgili sayfalarında inceleyebilirsiniz.

ÖRNEK: // GLDUSD ALTIN KİLO FİYATI

```
var GLDUSD = Sistem.SembolTanimla("DFN'GLDUSD", 2);
GLDUSD.Description = "ALTIN KİLO FİYATI";
GLDUSD.BidPrice = Convert.ToSingle((GLD.BidPrice)*31.99);
GLDUSD.AskPrice = Convert.ToSingle((GLD.AskPrice)*31.99);
GLDUSD.LastPrice = (GLDUSD.BidPrice+GLDUSD.AskPrice)/2;
Sistem.YuzeyselGuncelle(GLDUSD);
Sistem.GrafikGuncelle(GLDUSD);
```



- Grafik Verisi İndir – Sistem.GrafikVerisiIndir(Sembol,Periyot)

İstenilen bir sembolün istenilen bir periyodunun tüm geçmiş grafiğinin iDeal sunucularından indirilmesi için kullanılır. Bu fonksiyonu ardarda 1 saat geçmeden kullanamazsınız. Robot kodlarına bu fonksiyonu ekleyerek, çeşitli sebeplerle verisi eksik kalmış/olabilecek baz sembolünüzün verilerinin sürekli güncel kalmasını sağlarsınız. Kullanım / Yazım şekli aşağıdaki gibidir;

Sistem.GrafikVerisiIndir(Sembol, Periyot)

### Örnek:

var Sembol = "IMKBH'EREGL"; var Periyot = "60"; Sistem.GrafikVerisiIndır(Sembol, Periyot);

Veya direk tek satırda da yazılabilir; Sistem.GrafikVerisiIndir("VIP'VIP-X030","60");

Not: Fonksiyonu kod bloğunun en başına (henüz hesaplamaların yapılmadığı yere) yazmanızda yarar var.

### - Grafik Verisilerinde Tarih Hizalama – Sistem.GrafikVerisiIndir(Sembol,Periyot)

Farklı zaman dilimlerinde veri akışı olan piyasalara veya sembollere ait grafik verilerinde aynı zaman dilimlerinde barlar olmayabilir. Örneğin hisse piyasasında saat 10 ile 18:10 arasında işlemler olur ve bu aralıkta barlar oluşurken, VIOP tarafında işlemler 09:30 başlar. Yine aynı şekilde döviz kurlarında 5 gün 24 saat veri vardır. Eğer aynı zaman dilimlerinde aynı sayıda bar

içermeyen sembollere ait grafik verilerini birlikte görmek, çizmek veya kıyaslamalar yapmak istenirse, Sistem.GrafikVerilerindeTarihHizala komutu ile grafik barlarının aynı zaman dilimi için biribiyle hizalanması sağlanır.

Yazım şekli aşağıdaki gibidir;

Sistem.GrafikVerilerindeTarihHizala(Veriler1, Veriler2)

ÖRNEK: DOLAR/TL, ENDEKS100 VE VIOP ENDEKS grafiklerini (kapanışlarını) indikatör gibi başka bir grafiğin altında çizdirip takip ettirmek;

```
var SembolGrafik = Sistem.Sembol;
var Veriler = Sistem.GrafikVerileri;
var SembolVOB = "VIP'VIP-X030";
var VerilerVOB = Sistem.GrafikVerileriniOku(SembolVOB, Sistem.Periyot);
var BIST100 = "IMKBX'XU100";
var Veriler100 = Sistem.GrafikVerileriniOku(BIST100, Sistem.Periyot);
var DOLARTL = "FX'USDTRY";
var VerilerDOLAR = Sistem.GrafikVerileriniOku(DOLARTL, Sistem.Periyot);
// Verileri Hizala
VerilerVOB = Sistem.GrafikVerilerindeTarihHizala(Veriler, VerilerVOB);
Veriler100 = Sistem.GrafikVerilerindeTarihHizala(Veriler, Veriler100);
VerilerDOLAR = Sistem.GrafikVerilerindeTarihHizala(Veriler, VerilerDOLAR);
Sistem.GrafikVerilerindeTarihHizala(Veriler1, Veriler2)
var CVOB = Sistem.GrafikFiyatOku(VerilerVOB, "Kapanis");
var CBIST100 = Sistem.GrafikFiyatOku(Veriler100, "Kapanis");
var CDOLAR = Sistem.GrafikFiyatOku(Veriler100, "Kapanis");
Sistem.Cizgiler[0].Deger = CVOB; //Panel2
Sistem.Cizgiler[0].Aciklama = "VOB Close";
Sistem.Cizgiler[1].Deger = CBIST100; //panel3
Sistem.Cizgiler[1].Aciklama = "XU100 Close";
Sistem.Cizgiler[2].Deger = CDOLAR;
Sistem.Cizgiler[2].Aciklama = "USDTRY Close"; //panel4
```



### - Hacim - Sistem.Hacim(Sembol)

Bir sembolün o an ki anlık işlem hacmiyle birlikte, çeşitli dönemler için olan TL işlem hacmini okumak için kullanılır. Fonksiyonun içine Hacmi okutulmak istenen sembol yazılır.

<u>Not:</u> iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir.)

Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.HacimAltiAy(Sembol);
Sistem.HacimBirAy(Sembol);
Sistem.HacimBirHafta(Sembol);
Sistem.HacimBirYil(Sembol);
Sistem.HacimBuAy(Sembol);
Sistem.HacimBuHafta(Sembol);
Sistem.HacimGun(Sembol);
Sistem.HacimGun(Sembol);
Sistem.HacimSeans(Sembol);
Sistem.HacimUcAy(Sembol);
```

Örnek: GARAN hisse senedinin, bu doküman yazılırken ki son 1 yıllık hacim değerinin okuyup ekrana mesaj olarak çıkaran kod:

```
var YillikHacim = Sistem.HacimBirYil(Sembol);
Sistem.Mesaj("1 yıllık hacim = " + YillikHacim.ToString());
```



### - Hafta Sonu - Sistem.HaftaSonu

Sürekli çalışır durumda olan sistemlerde/robotlarda, olası hataların önüne geçmek üzere sunulan güvenlik amaçlı kullanılan bir fonksiyondur. Haftasonu (mesela dışardan grafik verisi import edildiğinde sistemler/robotlar sinyal üretmesin, emir göndermesin diye veya bilgisayar boşuna robotu tarayıp kaynak tüketmesin) olup olmadığı kontrol edilip, haftasonu ise bir işlem yapma gibi koda eklemeler yapılabilir.

\*\*Yazdığınız bir kodun tamamı veya belli bir kısmı (mesela sadece emir gönderen kısım) Bu fonksiyonla kontrole tabi tutulabilir.

**ÖRNEK:** 

```
if (Sistem.HaftaSonu == true)
Sistem.Mesaj("Şu an Hafta Sonundayız. Kod çalıştırılmayacak");
else
{
Kodunuzun tamamı burada yer alır
}
```



### - Hesap Kurum - Sistem.HesapKurum()

iDeal Portföy penceresi, birden fazla kuruma ait hesapların tanımlanabildiği ve aynı anda farklı farklı kurumlardaki hesaplara emir gönderilebildiği bir yapıya sahiptir. Portföy penceresinde sol üstteki kutuda seçili olan Kurum/Hesap aktif hesaptır. Aksi kodlanmamışsa robotlar, trend ve indikatörl alarmlarına bağlı emirler bu aktif hesaba gönderilir. Dileyen kullanıcılarımız formül yazarken, formüllerinin/robotlarının sadece bir aracı kurum için çalışmasını (veya tersi bir aracı kurum için çalışmamasını) ayarlayabilirler.

Bunun için Sistem.KurumHesap() komutuyla, portföy penceresindeki aktif (seçili) kurumun hangisi olduğu bilgisine ulaşılır.

ÖRNEK:

```
if (Sistem.HesapKurum() == "ABC YATIRIM")
{
    // FORMÜL SATIRLARI BURADA
}
```

```
else
{
    Sistem.Mesaj("Bu formül ABC YATIRIM için çalışmamaktadır.");
}
```

#### <u>HHLL - Sistem.HHLL(14)</u>

HHLL (HisghesHighLowestLow) isimli indikatörü hesaplayıp çağıran ideal sistem fonksiyonudur. 1 adet parametre alır ve varsayılan değeri 14'tür. HHV (HighesHighValue) ve LLV (LowestLowValue) seviyeleri (yani girilen parametre kadar bar için görülen EN YÜKSEK değer ile EN DÜŞÜK değer ARASINDAKİ FARKI bir indikatör olarak hesaplar çizer. Grafik barları üzerine değil, alt bölgedeki panellere çizilir. Kullanım ve yazım şekilleri aşağıdaki gibidir;

```
Sistem.HHLL(14);
Sistem.HHLL(14, Liste);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var L1 = Sistem.HHLL(14);
Sistem.Cizgiler[0].Deger = L1; //Panel no 2 yapılmalıdır
```

#### - HHV - Sistem.HHV(14)

HHV (Hisghest High Value) isimli indikatörü hesaplayıp çağıran ideal sistem fonksiyonudur. 1 adet parametre alır ve varsayılan değeri 14'tür. HHV (HighesHighValue) girilen parametre kadar bar için görülen EN YÜKSEK değeri bir indikatör olarak hesaplar çizer. Grafik barları üzerine çizdirilir ve böylece fiyat kesişimleri veya kanal analizi yapılabilir. Kullanım ve yazım şekilleri aşağıdaki gibidir;

```
Sistem.HHV(14);
Sistem.HHV(14, Liste);
Sistem.HHV(14, "Yuksek");
Sistem.HHV(GrafikVerileri, 14, "Yuksek"); //Başka bir sembole veya grafik periyoduna ait
barlar için HHV hesaplar.
```

Örnek: İndikatörü çağırıp çizdirmek

```
var HH = Sistem.HHV(14);
Sistem.Cizgiler[0].Deger = HH;
```

#### - Hisse İşlemlerini Oku - Sistem. HisselslemleriniOku()

Hisse senetlerinde ve varantlarda gerçekleşen işlemlerin tümünü (tüm işlemler) okuyarak kullanıcıya üzerinde hesaplama yapma imkanı veren ideal sistem fonksiyonudur. Eğer en az Düzey1+ (1 kademe derinlik) lisansınız varsa, gerçekleşen işlemlerin tamamını okutup, örneğin kademe analizi, aracı kurum analizi, net lot hesaplatma, bir aracı kurumun bir hissede yaptığı işlemleri baz alarak emir ileten robotlar yazma gibi formüller yazabilirsiniz.

Fonksiyon çalıştırıldığı zaman, geriye her bir gerçekleşen işleme ait veriler (işlemin saati/dakikası/saniyesi, işlemin fiyatı, işlemin lotu, alıcı kurum kodu, satıcı kurum kodu, işlem olduğu andaki en iyi alış fiyatı, işlem olduğu andaki en iyi satış fiyatı, işlemin borsa tarafından

üretilen ID (numarası) ve işlemin yönü (bir alış işlemi mi, bir satış işlemi mi) bilgilerine sahip olunur.

Fonksiyonun tüm dönüş bilgileri ve değişken tipleri aşağıda belirtilmiştir;

```
string Symbol = "";
byte Hour = 0;
byte Minute = 0;
byte Second = 0;
float Price = 0;
float Bid = 0;
float Ask = 0;
float Size = 0;
string BuyerCode = ""; //alıcı kurumun 3 harfli kısa kodudur.
string SellerCode = ""; //satıcı kurumun 3 harfli kısa kodudur.
string AggresiveParty = " "; // B,S (B-buy ise ALIŞ işlemidir, S-sell ise SATIŞ işlemidir)
string NasdaqTradeld = "";
```

Not1: Alan ve Satan kurum isimleri (kodları) PITE lisansı (anlık alan satan) olan kullanıcılar tarafından görülebilir.

Not2: Alan ve Satan kurum isimleri (kodları) c:\ideal\config klasörü altındaki Kurumlar.001 isimli dosya içerisinde yer almaktadur. Bu dosyayı NOTEPAD ile açarak hangi kurumunun kısa kodunun ne olduğu bilgisi görülebilir.

```
Örnek Kullanım-1: (GARAN hissesi için ilk 5 kurum seviyesinde PGÇ hesaplayan kod)
Not: kodun içindeki GARAN satırı değiştirilerek veya satırın kopyası alınıp yapıştırılıp başka
seneyler yazılarak, birden fazla senet grubu için de PGÇ hesaplatılabilir.
```

```
var Seviye = 5; //kaç kurum seviyesi için PGÇ
var Semboller = new Dictionary<string, bool>();
Semboller["GARAN"] = true;
//Yukarıya istenildiği kadar senet eklenebilir Bu durumda o senet grubunun TOPLAM PGC değeri elde
edilir.
//Kodun alt kısmında hiçbir değişiklik yapmanıza gerek yoktur.
var KurumlarNet = new Dictionary<string, double>();
var Islemler = Sistem.HisseIslemleriniOku();
foreach (var Islem in Islemler)
{
    if (Semboller.ContainsKey(Islem.Symbol))
    {
       if (KurumlarNet.ContainsKey(Islem.BuyerCode) == false)
           KurumlarNet[Islem.BuyerCode] = 0;
       if (KurumlarNet.ContainsKey(Islem.SellerCode) == false)
           KurumlarNet[Islem.SellerCode] = 0;
       double TL = Islem.Price * Islem.Size;
       KurumlarNet[Islem.BuyerCode] += TL;
       KurumlarNet[Islem.SellerCode] -= TL;
   }
}
KurumlarNet = KurumlarNet.OrderBy(x => x.Value).ToDictionary(x => x.Key, y => y.Value);
double AlanNet = 0;
double SatanNet = 0;
if (KurumlarNet.Count > 2 * Seviye)
```

```
{
    for (int i = 0; i < Seviye; i++)
        SatanNet += KurumlarNet.ElementAt(i).Value;
    for (int i = KurumlarNet.Count - Seviye; i < KurumlarNet.Count; i++)
        AlanNet += KurumlarNet.ElementAt(i).Value;
}
// PGC
double PGC = AlanNet + SatanNet;
Sistem.Mesaj("Para Giriş Çıkışı = " + PGC.ToString("0,000"));//ekranda mesaj olarak görmek için</pre>
```

Örnek Kullanım-2: CITI MENKUL kurumu ASELS hisse senedinde ALIŞ yapınca 1 lot ASELS alımı yapajn robot kodu

```
var Senet = "ASELS";
var Miktar = 1;
var Anahtar = Sistem.Name + DateTime.Now.Date.ToString("yyyyMMdd");
var TradeID = Sistem.SayiTablosunuOku(Anahtar);
var Emirler = Sistem.HisseIslemleriniOku(TradeID);
var Pozisyon = Sistem.PozisyonKontrolOku(Anahtar+Senet);
if (TradeID > 0 && Emirler.Count > 0)
{
   foreach (var item in Emirler)
   {
      if (item.Symbol == Senet && (item.BuyerCode == "CIM"))
      {
         Sistem.PozisyonKontrolGuncelle(Anahtar+Senet, Pozisyon+Miktar);
         Sistem.EmirSembol = "IMKBH'"+Senet;
         Sistem.EmirIslem = "Alış";
         Sistem.EmirMiktari = (int)Miktar;
         Sistem.EmirSuresi = "KIE"
         Sistem.EmirTipi = "Piyasa";
         Sistem.EmirGonder();
      }
     }
   }
TradeID = Emirler[Emirler.Count - 1].TradeID;
Sistem.SayiTablosunuGuncelle(Anahtar, TradeID);
```

- HY - Sistem. HY(Periyot)

Bir grafik üzerinde, girili periyot içerisinde görülen EN YÜKSEK değerden yüzdesel olarak ne kadar uzakta olduğumuz bilgisini dönen bir indikatör olan HY indikatörünü hesaplar. Bir adet parametre alır. Kullanım şekli Sistem.HY(20); şeklindedir.

Örnek: Momentum kendi ortalamasının üstündeyse, fiyatın 20 barlık en yükseğinden uzaklığı %2 den küçükse AL, tersi durumda SAT örnek sistemi.

```
var HY = Sistem.HY(20); // 20 periyottaki en yüksekle fiyat arasındaki yüzde fark
var Mom = Sistem.Momentum(5);
var AvrMom = Sistem.MA(Mom, "Simple", 10);
var SonYon = "";
for (int i=1; i < Sistem.BarSayisi; i++)
{
    HY[i] = Math.Abs(HY[i]);
    if (SonYon != "A" && HY[i] < 2 && Mom[i] > AvrMom[i])
    {
        SonYon = "A";
        Sistem.Yon[i] = "A";
    }
}
```

```
}
if (SonYon != "S" && HY[i] > 2 && Mom[i] < AvrMom[i])
{
    SonYon = "S";
    Sistem.Yon[i] = "S";
}</pre>
```

#### - Ichi Moku - Sistem.Ichimoku()

Bir grafik üzerinde Ichi Moku olarak bilinen indikatörü hesaplar. Kullanım şekli Sistem.Ichimoku(); şeklindedir. Bu fonksiyon tek başına kullanıldığı zaman herhangi bir görsel efekt yaşanmaz grafik üzerinde veya robotlarda. Fonksiyon çağrıldığı zaman geriye bir nesne döner. Bu nesnenin içinde Ichi Moku indikatörünün 5 adet listesi (çizgisi) vardır. Bu listelere (çizgilere) Ichi Mokuyu tanımlarken kullanılmış olan değişken isminden sonra noktaya basılarak ilgili çizginin adı yazılarak erişilir.

Kullanım şu şekildir;

```
var IchiMoku = Sistem.Ichimoku(); //Ichimoku indikatörüne ait çizgileri çağıran nesne
var Tenkansen = IchiMoku.Tenkansen;
var Kijunsen = IchiMoku.Kijunsen;
var ChikouSpan = IchiMoku.ChikouSpan;
var SenkouSpanA = IchiMoku.SenkouSpanA;
var SenkouSpanB = IchiMoku.SenkouSpanB;
```

Örnek: Fiyat hem Tenkansaen hem de Kijunsen üzerindeyse AL tersi durumda SAT sinyali üreten sistem formülü

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var IchiMoku = Sistem.Ichimoku();
var Tenkansen = IchiMoku.Tenkansen;
var Kijunsen = IchiMoku.Kijunsen;
var SonYon = "";
for(int i=1; i < Sistem.BarSayisi;i++)</pre>
{
   if (C[i] > Tenkansen[i] && C[i] > Kijunsen[i] && SonYon != "A")
   {
      SonYon = "A";
      Sistem.Yon[i] = "A";
   }
   else if (C[i] < Tenkansen[i] && C[i] < Kijunsen[i] && SonYon != "S")
   {
      SonYon = "S";
      Sistem.Yon[i] = "S";
   }
}
```

#### - IFISH CCI - Sistem. IFISHCCI(6, 9)

CCI indikatörünü kullanarak hesaplatılan IFISH isimli indikatörü hesaplayıp çağıran sistem fonksiyonudur.

Kullanım şekli aşağıdaki gibidir;

Sistem.IFISHCCI(6, 9);

Örnek: İndikatörü çağırıp çizdirmek



var x = Sistem.IFISHCCI(6, 9); Sistem.Cizgiler[0].Deger = x; // 2 nuömaralı panele çizdirilir.

- IFISH RSI - Sistem. IFISHRSI(6, 9)

RSI indikatörünü kullanarak hesaplatılan IFISH isimli indikatörü hesaplayıp çağıran sistem fonksiyonudur.

Kullanım şekli aşağıdaki gibidir;

Sistem.IFISHRSI(6, 9);

Örnek: İndikatörü çağırıp çizdirmek

```
var x = Sistem.IFISHRSI(6, 9);
Sistem.Cizgiler[0].Deger = x; // 2 nuömaralı panele çizdirilir.
```

- IMI / Intraday Momentum Index - Sistem.IMI(14)

IMI (Intraday Momentum Index) olarak bilinen indikatörü hesaplayıp çağıran sistem fonksiyonudur. Varsayılan olarak 14 parametresi alır. Kullanım şekli aşağıdaki gibidir;

```
Sistem.IMI(14);
Sistem.IMI(Veriler, 14);
```

Örnek: İndikatörü çağırıp çizdirmek (indikatör üzerinde yatay 30 ve 70 seviyelerine de birer çizgi çizdirilmiştir.)

```
var IMI = Sistem.IMI(14);
Sistem.Cizgiler[0].Deger = IMI; // 2 nuömaralı panele çizdirilir.
Sistem.Cizgiler[1].Deger = Sistem.Liste(30); // 2 nuömaralı panele çizdirilir.
Sistem.Cizgiler[2].Deger = Sistem.Liste(70); // 2 nuömaralı panele çizdirilir.
```

- İzleyen Stop Puan - Sistem. IzleyenStopPuan(2,i)

Grafikler üzerinde Al ve Sat sinyalleri üreten bir strateji/sistem formülü yazıldığı zaman, ALIM veya SATIM pozisyonuna girildiği fiyata PUAN (TL) bazında izleyen (iz süren) stop eklemek için kullanılan ideal sistem fonksiyonudur. Fonksiyon içerisine yazılmış kadar puan ile izleyen; AL yönündeyken bu fiyat seviyesi aşağı yönde kesildiğinde veya SAT yönündeyken bu fiyat seviyesini yukarı yönde kesildiğinde sistemi FLAT (Nakit) pozisyona çekmek için kullanılır. Kullanım şekli aşağıdaki gibibidir;

```
Sistem.IzleyenStopPuan(2,i);
```

Örnek: IMI, RSI ve MOV kullanılarak üretilmiş örnek bir AL/SAT sistemi ve bu sisteme VIOP ENDEKS için 1200 puan izleyen stop ekleme formülü

```
var V = Sistem.GrafikVerileri;
var C = Sistem.GrafikFiyatSec("Kapanis");
var H = Sistem.GrafikFiyatSec("Yuksek");
var L = Sistem.GrafikFiyatSec("Yuksek");
```

```
var XX = C;
var X1 = Sistem.MA(XX, "Exp", 10);
var X2 = Sistem.MA(XX, "Exp", 20);
var YY = Sistem.RSI(10);
var Y1 = Sistem.MA(YY, "Exp", 20);
var Y2 = Sistem.MA(YY, "Exp", 40);
var ZZ = Sistem.IMI(7);
var Z1 = Sistem.MA(ZZ, "Exp", 13);
var Z2 = Sistem.MA(ZZ, "Exp", 36);
var KARAL = Sistem.Liste(0);
var IZLEYENSTOP = Sistem.Liste(0);
var SonYon = "";
var FlatOncesiYon = "";
for (int i = 1 ; i < Sistem.BarSayisi; i++)</pre>
{
   var IndikatorAlis = X1[i] > X2[i] && Y1[i] > Y2[i] ;
   var IndikatorSatis = X1[i] < X2[i] && Y1[i] < Y2[i];</pre>
   IZLEYENSTOP[i] = Sistem.IzleyenStopPuan(1.200, i); //1200 puan izleyen stop
   if (IZLEYENSTOP[i] == 0) IZLEYENSTOP[i] = C[i];
   if (C[i] < IZLEYENSTOP[i] && SonYon == "A")</pre>
   {
       FlatOncesiYon = SonYon;
      SonYon = "F";
      Sistem.Yon[i] = "F";
   }
   else if ((C[i] > IZLEYENSTOP[i] || C[i] <= KARAL[i]) && SonYon == "S") // satistan flate
   {
      FlatOncesiYon = SonYon;
      SonYon = "F";
      Sistem.Yon[i] = "F";
   }
   else if (IndikatorAlis && SonYon != "A" && FlatOncesiYon != "A") // alış
   {
        FlatOncesiYon = "";
        SonYon = "A";
        Sistem.Yon[i] = "A";
   }
   else if (IndikatorSatis && SonYon != "S" && FlatOncesiYon != "S") // satış
   {
       FlatOncesiYon = "";
       SonYon = "S";
       Sistem.Yon[i] = "S";
   }
}
Sistem.Cizgiler[0].Deger = IZLEYENSTOP;
Sistem.Cizgiler[1].Deger = X2;
Sistem.Cizgiler[2].Deger = IZLEYENSTOP;
Sistem.Cizgiler[3].Deger = C;
```

- İzleyen Stop Yüzde - Sistem. IzleyenStopYuzde(1.5,i)

Grafikler üzerinde Al ve Sat sinyalleri üreten bir strateji/sistem formülü yazıldığı zaman, ALIM veya SATIM pozisyonuna girildiği fiyata, YÜZDE (%) bazda izleyen (iz süren) stop eklemek için kullanılan ideal sistem fonksiyonudur. Fonksiyon içerisine yazılmış kadar yüzde ile izleyen; AL yönündeyken bu fiyat seviyesi aşağı yönde kesildiğinde veya SAT yönündeyken bu fiyat seviyesini yukarı yönde kesildiğinde sistemi FLAT (Nakit) pozisyona çekmek için kullanılır. Kullanım şekli aşağıdaki gibibidir;



```
Sistem.IzleyenStopPuan(2,i);
```

## Örnek: 2 hareketli ortalamanın kesişimine göre ALIM yapan sisteme %1.5 izleyen stop eklenmesi

```
var IzleyenYuzde = 1.5; //%1.5 izleyen stop
var C = Sistem.GrafikFiyatSec("Kapanis");
var SonYon = "";
// hareketli ortalamalari hesapla
var MA1 = Sistem.MA(Veriler, "Exp", 10);
var MA2 = Sistem.MA(Veriler, "Exp", 100);
var IZLEYENSTOP = Sistem.Liste(0);
for (int i = 1; i < Veriler.Count; i++)</pre>
{
   IZLEYENSTOP[i] = Sistem.IzleyenStopYuzde(IzleyenYuzde, i);
   if (IZLEYENSTOP[i] == 0) IZLEYENSTOP[i] = Veriler[i];
   if (MA1[i-1] < MA2[i-1] && MA1[i] > MA2[i] && SonYon!="A" ) // 1.ortalama 2.ortalamanın üstüne
cikarsa
   {
      Sistem.Yon[i] = "A"; //AL
      SonYon = "A";
   }
   else if (C[i] < IZLEYENSTOP[i] && SonYon!="F" ) // İzleyen Stop ile nakite geç
   {
      Sistem.Yon[i] = "F"; // SAT
      SonYon = "F";
   }
}
```

### - Kademe Analizi Oku – Sistem.KademeAnalizOku(Sembol)

Veri terminallerinde yatırımcıların en çok takip ettiği analizlerden biri Kademe Analizidir. Bir hissede veya vadeli sözleşmede hangi fiyattan kaç adet işlem olmuş, bu işlemlerin ne kadarı alıştan, ne kadarı satıştan olmuş, ortalama fiyat nedir gibi bilgileri bu analiz sunmaktadır. Sistem fonksiyonlarını kullanarak bir hissenin, vadeli sözleşmenin, varantın veya opsiyonun, o gün o ana kadar gerçekleşmiş işlemlerinden oluşan kadeame analizi verilerine erişmeyi sağlayan sistem fonksiyonudur. Aşağıda gösterildiği gibi, tüm günün, belli bir saat aralığının ve son x dakikanın kademe analizini okumaya yönelik 3 farklı kullanım şekli vardır;

```
Sistem.KademeAnalizOku(Sembol);//tüm günün kademe analizi okunur
Sistem.KademeAnalizOku(Sembol, "10:30", "11:45");//girilen iki saat arasındaki kademe
analizini okur
Sistem.KademeAnalizOku(Sembol, 15); //Son 15 dakikanın kademe analizini okur
```

Kademe Analizi oku fonksiyonu ile kademeler okutulup bir değişkene atandığında (yani var x = Sistem.KademeAnalizOku(Sembol); satırı yazıldığında) atanmış değişken içinde birçok veriyi barındıran bir nesne/obje olarak sunulmuş olur. Bu nesne içindeki bilgilerden herhangi birine x.BilgiAdi şeklinde yazarak erişiriz.

Örnek: GARAN hisse senedinin Kademe Analizini okuyup, hissede toplam kaç TL işlem olmuş olduğu verisine erişelim;

```
var Sembol = "IMKBH'GARAN";
var Kademeler = Sistem.KademeAnalizOku(Sembol);
var Hcm = Kademeler.ToplamHacim;
Sistem.Mesaj(Hcm.ToString("0,000"));
```

Formülü yazıp kaydettikten sonra Formül Test butonuna basarsak, GARAN hisse senedinde o ana kadar gerçekleşmiş işlemlerin toplam hacim (TL) verisini okur ve ekrana bir mesaj penceresi açarak gösterir.

Dikkat edilirse, kademe analizi oku komutu ile çağrılan kademe analizi verileri ismi **Kademeler** olan bir değişkene geliyor. Bu değişken (içinde bir çok bilgi içeren bir nesnedir) içinden mesela Toplam Hacim verisini okumak için değişken adının sonuna nokta işareti koyup toplam hacmi veren bilgi alanının adını yazıyoruz.

```
var Sembol = "IMKBH'GARAN";
var Kademeler = Sistem.KademeAnalizOku(Sembol);
var Hcm = Kademeler.ToplamHacim;
Sistem.Mesaj(Hcm.ToString("0,000"));
```

🖳 iDeal Mesaj 08.07.2020 12:51:36

680,212,770

Kademe Analizi Oku denildiği zaman elde edilen nesne içinde yer alan bilgilerin tamamı decimal sayı formatındadır. Bilgiler ve yazılış şekilleri aşağıdadır. (kademe analizi oku denildiğinde aşağıdaki bilgiler elde edilebilir)

Yukarıda belirtilen kullanım şekillerinde de gösterildiği gibi, istenirse kullanıcı tarafından girilecek 2 saat arasındaki işlemlerin veya bulunduğumuz saat/dakikaya göre son x dk olan işlemlerin kademe analizi verileride hesaplatılıp kullanılabilir.

```
decimal ToplamHacim = 0;
decimal ToplamLot = 0;
decimal Ortalama = 0;
decimal AlisLot = 0;
decimal SatisLot = 0;
decimal MaxToplamLot = 0;
decimal MaxAlisLot = 0;
decimal MaxSatisLot = 0;
decimal Fark = 0;
decimal MaxAlisFark = 0;
decimal MaxSatisFark = 0;
decimal MaxSatisFark = 0;
decimal MaxSatisLotFiyat = 0;
decimal MaxAlisLotFiyat = 0;
```

Örnek-2: ASELSAN hisse senedinin kademe analizini okuyan, bu analizde sunulan tüm bilgi alanlarına erişip, o bilgileri ekrana açtırılacak bir tabloya yazan kod örneği;

```
var Sembol = "IMKBH'ASELS";
var Kademeler = Sistem.KademeAnalizOku(Sembol);
var Hcm = Kademeler.ToplamHacim;
```



var SatisLot = Kademeler.SatisLot; var AlisLot = Kademeler.AlisLot; var MaxLot = Kademeler.MaxToplamLot; var MaxSatisHcm = Kademeler.MaxSatisLot; var MaxAlisHcm = Kademeler.MaxAlisLot; var Delta = Kademeler.Fark; var DeltaMax = Kademeler.MaxAlisFark; var DeltaMin = Kademeler.MaxSatisFark; var VWAP = Kademeler.Ortalama; var MacKademeFiyati = Kademeler.MaxToplamLotFiyat; // Tablo Yarat string TabloAd = Sembol + " " + "Kademe Analizi Detaylar1 "; //ISIM VER var SutunGenislik = new int[2] {300,100}; //SÜTUN SAYISI VE SÜTUN GENİŞLİKLERİ var SutunHizala = new int[2] { 0,2}; //HİZALAMA (1=ORTALA) var SutunBaslik = new string[2] { "Veri", "Değer"}; //BAŞLIKLAR Sistem.Tablo(TabloAd, 100, 200, 480, 350, 2, 15, SutunGenislik, SutunHizala, SutunBaslik, 1, 8); //TABLOYU EKRANA GETİR //Tabloyu Doldur Sistem.TabloYazdir(TabloAd, 1, 0, Hcm.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 0, "Toplam İşlem Hacmi", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 1, SatisLot.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 1, "Toplam Satistan Lot", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 2, AlisLot.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 2, "Toplam Alıştan Lot", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 3, MaxLot.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 3, "En Çok işlem Olan Kademenin Lotu", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 4, MaxSatisHcm.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 4, "En cok satistan lot", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 5, MaxAlisHcm.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 5, "En cok alıştan lot", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 6, Delta.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 6, "Alış Satış Farkı", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 7, DeltaMax.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 7, "Alış Satış Farkı En Büyük Olan Kademe Lotu", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 8, DeltaMin.ToString("0,000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 8, "Satış Alış Farkı En Büyük Olan Kademe Lotu", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 9, VWAP.ToString("0.0000"), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 9, "Ağırlıklı Ortalama", Color.White, Color.Red); Sistem.TabloYazdir(TabloAd, 1, 10, MacKademeFiyati.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 10, "En Çok işlem olan kademenin Fiyatı", Color.White, Color.Red);

IMKBH'ASELS Kademe Analizi Detayları		
Veri	Değer	
Toplam İşlem Hacmi	313,922,740	
Toplam Satıştan Lot	4,962,756	
Toplam Alıştan Lot	4,656,751	
En Çok işlem Olan Kademenin Lotu	1,062,835	
En çok satıştan lot	711,442	
En çok alıştan lot	653,828	
Alış Satış Farkı	-306,005	
Alış Satış Farkı En Büyük Olan Kademe Lotu	248,266	
Satış Alış Farkı En Büyük Olan Kademe Lotu	-370,944	
Ağırlıklı Ortalama	32.6340	
En Çok işlem olan kademenin Fiyatı	32.72	

### - KAİRİ - Sistem.Kairi(14)

KAIRI isimli indikatörü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi 3 kullanım şekli vardır;

```
Sistem.Kairi(14);
Sistem.Kairi(Liste,14);
Sistem.Kairi(Veriler,14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var K = Sistem.Kairi(14);
Sistem.Cizgiler[0].Deger = K;
```

- KAMA - Sistem.KAMA(10,2,30)

KAMA isimli indikatörü çağırır. 3 adet parametre (periyot) alır ve varsayılan değerler olarak 10, 2 ve 30 kullanılır. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.KAMA(10, 2 , 30);
Sistem.KAMA(Veriler,10, 2 , 30);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var Kama = Sistem.KAMA(10, 2 , 30);
Sistem.Cizgiler[0].Deger = Kama;
```

```
    KAR AL PUAN - Sistem.KarAlPuan(3,i)
```

Grafikler üzerinde Al ve Sat sinyalleri üreten bir strateji/sistem formülü yazıldığı zaman, ALIM veya SATIM pozisyonuna girildiği fiyata PUAN (TL) bazında KAR AL eklemek için kullanılan ideal sistem fonksiyonudur. AL yönünde sinyal geldiği zaman, işleme girilen fiyat (aksi

belirtilmedikçe al sinyali olan barın kapanış değeridir) üzerine, bu fonksiyonun içine yazılan kadar puan/TL eklenir. Fiyat bu yeni fiyatın (işleme giriş fiyatı + karal puan değeri) üstüne çıktığı zaman Sistem KAR AL eylemi yapar, SATIŞ işlemi yapılarak pozisyon kapatılır ve nakite geçilir. Aynı şekilde Sat (açığa satış veya short) pozisyonundayken de işleme giriş fiyatından karal puanı kadar aşağıdaki fiyat seviyesinin altına inince açığa satış veya short pozisyon ALIŞ işlemi yapılarak kapaılıp KAR AL yapılmış olur. Kullanım şekli aşağıdaki gibibidir;

Sistem.KarAlPuan(0.06,i); (Not: Garan için alış fiyatı 8.22 ise ve 6 kademe yukarıya (yani 8.28 fiyata) kar al yazılacaksa, bu durumda kar al içine yazılacak puan 8.28-8.22 = 0.06 dır)

#### KAR AL YÜZDE - Sistem.KarAlYuzde(3,i)

Grafikler üzerinde Al ve Sat sinyalleri üreten bir strateji/sistem formülü yazıldığı zaman, ALIM veya SATIM pozisyonuna girildiği fiyata % bazında KAR AL eklemek için kullanılan ideal sistem fonksiyonudur. AL yönünde sinyal geldiği zaman, işleme girilen fiyatın (aksi belirtilmedikçe al sinyali olan barın kapanış değeridir) bu fonksiyonun içine yazılan kadar yüzde yukarısındaki seviyeye çıktığı zaman Sistem KAR AL eylemi yapar, SATIŞ işlemi yapılarak pozisyon kapatılır ve nakite geçilir. Aynı şekilde Sat (açığa satış veya short) pozisyonundayken de işleme giriş fiyatından karal yüzdesi kadar aşağıdaki fiyat seviyesinin altına inince açığa satış veya short pozisyon ALIŞ işlemi yapılarak kapaılıp KAR AL yapılmış olur. Kullanım şekli aşağıdaki gibibidir;

```
Sistem.KarAlYuzde(2.40,i);
(Not: İşleme girilen fiyatın %2.4 kadar üstünde (satış yönü için altında) fiyat seviyesi görülünce
kar alma eylemi gerçekleşir.)
```

Örnek: KAR AL YÜZDE uygulanmış örnek bir sistem formülü

```
var V = Sistem.GrafikVerileri;
var C = Sistem.GrafikFiyatSec("Kapanis");
var XX = C;
var X1 = Sistem.MA(XX, "Exp", 10);
var X2 = Sistem.MA(XX, "Exp", 50);
var YY = Sistem.RSI(100);
var Y1 = Sistem.MA(YY, "Exp", 10);
var Y2 = Sistem.MA(YY, "Exp", 200);
var ZZ = Sistem.IMI(80);
var Z1 = Sistem.MA(ZZ, "Exp", 10);
var Z2 = Sistem.MA(ZZ, "Exp", 200);
var KARAL = Sistem.Liste(0);
var SonYon = "";
var FlatOncesiYon = "";
var FlatFiyat = 0.0f;
for (int i = 1 ; i < Sistem.BarSayisi; i++)</pre>
{
   var IndikatorAlis = X1[i] > X2[i] && Y1[i] > Y2[i] && Z1[i] > Z2[i];
   var IndikatorSatis = X1[i] < X2[i] && Y1[i] < Y2[i] && Z1[i] < Z2[i];</pre>
   KARAL[i] = Sistem.KarAlYuzde(3.6,i);
   if (KARAL[i] == 0) KARAL[i] = C[i];
   if (SonYon == "F" && FlatOncesiYon == "A" ) // kar alındıktan sonra tekrar alış trendine girme
```

```
{
      FlatOncesiYon = "";
      SonYon = "A";
      Sistem.Yon[i] = "A";
   }
   else if (SonYon == "F" && FlatOncesiYon == "S") // kar alındıktan sonra tekrar alış trendine
girme
   {
      FlatOncesiYon = "";
      SonYon = "A";
      Sistem.Yon[i] = "A";
   }
   else if (C[i] >= KARAL[i] && SonYon == "A") // aliştan KAR AL
   {
      FlatOncesiYon = SonYon;
      FlatFiyat = C[i];
      SonYon = "F";
      Sistem.Yon[i] = "F";
   }
   else if (C[i] <= KARAL[i] && SonYon == "S") // satıştan KAR AL
   {
      FlatOncesiYon = SonYon;
      FlatFiyat = C[i];
      SonYon = "F";
      Sistem.Yon[i] = "F";
   }
   else if (IndikatorAlis && SonYon != "A" && FlatOncesiYon != "A") // alış/LONG
   {
        FlatOncesiYon = "";
        SonYon = "A";
        Sistem.Yon[i] = "A";
   }
   else if (IndikatorSatis && SonYon != "S" && FlatOncesiYon != "S") // sati$/SHORT
   {
       FlatOncesiYon = "";
       SonYon = "S";
       Sistem.Yon[i] = "S";
   }
}
```

### - Kar Zarar İşlem - Sistem.KarZararIslem(IslemSayisi, KaymaMaliyeti)

Grafikler üzerinde Al ve Sat sinyalleri üreten bir strateji/sistem formülü yazıldığı zaman, bu sistemin geçmişten itibaren hesaplanan getiri/performans sonuçlarını kullanan, kullanıcı tarafından girilen kadar işlemin/sinyalin karzarar değerini indikatör olarak hesaplayıp çizen iDeal Sistem fonksiyonudur. Bir sistem yazıp grafiğe uygulayan bir kullanıcı, sisteminin getiri eğrisini alt bölgede indikatör olarak çizdirip, istersen mesela son 10 sinyalin getirisini de ayrı bir indikatör olarak hesaplatıp çizdirebiliriz. Bu son 10 sinyalin getirisini işlemlerden dolayı oluşacak kayma+komisyon maliyetini de düşerek hesaplatma imkanı da vardır. Fonksiyon bu dokumanın yukarıdaki sayfalarında anlatılan GETİRİ HESAPLA fonksiyonu ile birlikte kullanılır. Kullanım şekli aşağıdaki örnekte yer almaktadır.

Örnek: RS, MACD ve MOMENTUM indikatörlerine göre al-sat yapan bir örnek sistem ve o sistemin altında hem tüm sinyaller için hem de sadece son 5 işlemin/sinyalin getiri eğrisini çizen formül:

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var RSI = Sistem.RSI(C,14);
var MAV = Sistem.MA(600, "Simple", "Kapanis");
var M = Sistem.MACD(12, 26);
```

```
var SonYon = "";
for (int i = 4; i < Sistem.BarSayisi - 3; i++)
{
    if ( RSI[i] > 70f && M[i] > 0.10f && C[i] > MAV[i]-0.500f && SonYon != "A") // AL
        {
            Sistem.Yon[i] = "A";
            SonYon = Sistem.Yon[i];
        }
        else if (RSI[i] < 60f && M[i] < -0.10f && C[i] < MAV[i]+0.500f && SonYon != "S") // SAT
        {
            Sistem.Yon[i] = "S";
            SonYon = Sistem.Yon[i];
        }
    }
    Sistem.GetiriHesapla("01/01/2015",0.0);
    Sistem.Cizgiler[0].Deger = Sistem.GetiriKZ; //Panel 2
```

var KarZararIslemSayisiBazinda = Sistem.KarZararIslem(5, 0.0); Sistem.Cizgiler[1].Deger = KarZararIslemSayisiBazinda ; //panel3



### - Keltner Kanalı - Sistem.KeltnerUp/KeltnerDown(10)

Alt ve Üst olarak adlandırılan 2 çizgiden oluşan Keltner isimli isimli indikatörü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 10 kullanılır. Kanalın/bandın üst çizgisi için ayrı (UP), alt çizgisi için ayrı (DOWN) birer fonksiyon vardır. Aşağıdaki gibi kullanım şekli vardır;

```
Sistem.KeltnerDown(10);
Sistem.KeltnerUp(10);
Sistem.KeltnerDown(Veriler, 10);
Sistem.KeltnerUp(Veriler ,10);
```

Örnek: İndikatörün Alt ve Üst bandını çağırıp çizdirmek



var Alt = Sistem.KeltnerDown(10); var Ust = Sistem.KeltnerUp(10);

Sistem.Cizgiler[0].Deger = Alt; Sistem.Cizgiler[1].Deger = Ust;



### - Kesişme Tara - Sistem.KesismeTara(x,y)

En popüler teknik analiz stratejilerinden biri de iki çizgini (iki indikatörün birbiriyle veya bir indikatörün bir sabit değerle) kesişmesi yoluyla AL/SAT sinyalleri tespit etmektir.

Kesişme tespitleri yapmak ve buna göre AL/SAT sinyalleri üretmek için iDeal **Sistem.KesismeTara** fonksiyonunu sunmaktadır.

Bu fonksiyon, iki çizginin, birbirini her iki yönde de kesmesine göre strateji üretir.

**NOT**: Bir çizginin, bir diğer çizgiyi veya değeri, **sadece tek yönde** (Yukarı veya Aşağı yönde) kesmesine dayalı bir strateji için ayrıca iki fonksiyon (Bkz: **Sistem.YukariKestiyse** ve **Sistem.AsagiKestiyse**) mevcuttur.

**Sistem.KesismeTara** fonksiyonu, mutlaka belirtilmesi gereken 2 adet parametreye ihtiyaç duyar ve iki farklı şekilde kullanılabilir.

Sistem.KesismeTara(Cizgi1, Cizgi2); (veya Liste1/Liste2) Sistem.KesismeTara(Cizgi, SabitDeger);

Örneğin: İki farklı hareketli ortalamanın birbiriyle kesişimi veya RSI indikatörünün kendi ortalamasıyla kesişimi gibi bir strateji kullanılacaksa birinci şekilde yazılır.

#### - Sistem.KesismeTara(MA1, MA2);

Örneğin RSI indikatörünün sabit bir değeri (50 yi geçerse ) kesmesi stratejisi kullanılacaksa ikincisi kullanılır.

- Sistem.KesismeTara(RSI,50);

Kesişme Tara, Yukarı Kestiyse ve Aşağı Kestiyse fonksiyonları, kapanmış olan son iki barıdaki değerleri kıyaslayarak sonuç döndürürler. Henüz kapanmamış olan bar hesaplamaya/kontrole dahil edilmez.

Örneğin aşağıdaki grafikte, 1 ile numaralandırılan barda 10'luk ortalama, 35'lik ortalamanın üzerinde.

2 ile numaralandırılan bar kapanışında ise, altına iniyor (aşağı kesiyor.) sonraki ilk barda sinyal üretiliyor.



ÖRNEK: MACD indikatörünün kendi hareketli ortalamasıyla kesişimlerine dayalı bir strateji kodu:

var MACD = Sistem.MACD (12, 26); var AVR = Sistem.MA(MACD, "Exp", 9);

Sistem.Cizgiler[0].Deger = MACD; Sistem.Cizgiler[1].Deger= AVR;

// strateji
Sistem.KesismeTara(MACD, AVR);



#### - Klinger Oscilator - Sistem.KlingerOsc(13)

Klinger Oscilator isimli indikatörü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 13 kullanılır. Aşağıdaki gibi 2 kullanım şekli vardır;

Sistem.KlingerOsc(13); Sistem.KlingerOsc(Veriler,13);

Örnek: İndikatörü çağırıp çizdirmek

```
var X = Sistem.KlingerOsc(13);
Sistem.Cizgiler[0].Deger = X;
```

- Kurum Hacim Oku - Sistem.KurumHacimOku("IYF")

Aracı kurumların, geçmiş günlere ait hacimlerini ve tüm kurumların hacim toplamları içinde o kurumun hacminin yüzdesel oranını okumak için kullanılan iDeal Sistem fonksiyonudur. Fonksiyon çağrıldığı zaman içerisinde "Hacim" ve "Yuzde" olarak iki bilgi alanı içeren bir nesne (obje) döner. Objeyi tanımladığınız değişken isminin yanına "nokta" işareti koyup Hacim yazarak, seçilen kurumun gün gün işlem hacmi verisine, noktadan sonra "Yuzde" yazarak her bir gün için o kurumun hacminin toplam (tüm kurumların hacim toplamları) hacime yüzdesel oranı verisine erişilir.

Örnek kullanım şekli için aşağıda bir kod paylaşılmıştır. Bu kod AK YATIRIM ve INFO YATIRIM kurumlarının tarihsel işlem hacimlerini ve hacim oranlarını çağırıp birer indikatör olarak çizdirmektedir.

var AKYATIRIM = Sistem.KurumHacimOku("AKM");//AK Yatırım var INFOYATIRIM = Sistem.KurumHacimOku("IYF");//info yatırım

Sistem.Cizgiler[0].Deger = AKYATIRIM.Hacim; //Ak Yatırım hacim Sistem.Cizgiler[1].Deger = AKYATIRIM.Yuzde; //Ak Yatırım yüzde

Sistem.Cizgiler[2].Deger = INFOYATIRIM.Hacim; //info hacim Sistem.Cizgiler[3].Deger = INFOYATIRIM.Yuzde; //info yuzde



Kurum Hacim Oku fonksiyonu ile ilgili bilinmesi gereken bazı önemli detaylar aşağıdaki gibidir;

- Fonksiyon yazılırken parantez içinde (ve çift tırnak işaretleri arasında) hacim ve yüzde verisine ulaşılmak istenen Aracı Kurumun borsada ve Takasbank'ta kayıtlı olan 3 harfli kısa kodu yazılır.
- Hangi aracı kurumun kısa kodunun ne olduğunu öğrenmek için C:\iDeal\Config klasörü altında bulunan Kurumlar.001 isimli dosyanın içine bakılabilir. (Bu dosya not defteri ile açılabilir)
- Hacim Oku fonksiyonu c:\ideal\HacimImkb isimli klasörde tutulan dosyalardan veri okur. O klasörde hangi tarihe kadar dosya varsa, o kadar veriokutulur. Hacim verisi yok veya eksik ise, iDeal'in üst kısmındaki BIST menüsü altında DİĞER ANALİZLER satırına gelinip HACİMLER tablosu açılır. İstenen iki tarih arası seçilir ve GÜNCELLE butonuna basılarak girilmiş tarih aralığına ait hacim dosyaları iDeal sunucularından indirilir.
- Linear Regresyon Sistem.LinearReg(14)

Linear Regresyon ismiyle bilinen indikatörü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi 3 kullanım şekli vardır;

```
Sistem.LinearReg(14);
Sistem.LinearReg(Liste,14);
Sistem.LinearReg (Veriler,14);
```



Örnek: İndikatörü çağırıp çizdirmek

var LG = Sistem.LinearReg(14); Sistem.Cizgiler[0].Deger = LG;

- Linear Regresson Inter Cept - Sistem.LinearRegInterCept(14)

Linear Regresyon Inter Cept ismiyle bilinen indikatörü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi 2 kullanım şekli vardır;

```
Sistem.LinearRegInterCept(14);
Sistem.LinearRegInterCept(Liste,14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var LG = Sistem.LinearRegInterCept(14);
Sistem.Cizgiler[0].Deger = LG;
```

- Linear Regresyon Slope- Sistem.LinearRegSlope(14)

Linear Regresyon Slope ismiyle bilinen indikatörü çağırır. 1 adet parametre (periyot) alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi 3 kullanım şekli vardır;

```
Sistem.LinearRegSlope(14);
Sistem.LinearRegSlope(Liste,14);
Sistem.LinearRegSlope(Veriler,14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var LG = Sistem.LinearRegSlope(14);
Sistem.Cizgiler[0].Deger = LG;
```

- <u>Lisans Kontrol - Sistem.LisansKontrol("KullaniciAdi")</u>

iDeal üzerinden yazılmış bir sistemi/robotu/indikatörü bir veya daha fazla IDEAL KULLANICI ADI için lisanslayabilir, belirtilenlerden başka kullanıcı adıyla iDeal kullananların, dosya olarak elde etseler bile bu formülü kullanmalarına engel olabilirsiniz. Tabi bu fonksiyonu kullanan bir kullanıcının, formülünü ŞİFRELİ hale getirmesi gerekiyor.

**ÇOK ÖNEMLİ NOT**: Formül yazılan penceredeki ŞİFRELE butonuna bastığınız zaman, IDEAL rastgele bir algoritma ile kodu karıştırıp gizleyecektir. *Kodun açık halini, kullanıcının kendisi* <u>de bir daha göremeyecektir</u>. Bu nedenle, bir başkasıyla paylaşmak için şifrelenecek olan formülün önce bir başka isimle kopyasını oluşturup, bu kopyayı şifrelemeniz gerekmektedir.

```
ÖRNEK: //AHMET ve Sezai12 kullanıcılarına kullanım izni verilen, onun dışında bir username ile
iDeal'e bağlanan bir kişiye Bu Formülü Kullanma Yetkiniz Yoktur mesajı çıkaran ve çalışmayan kod
örneği
```

```
if (Sistem.LisansKontrol("AHMET","Sezai12"))
{
```

```
var Veriler = Sistem.GrafikFiyatSec("Kapanis");
var MA1 = Sistem.MA(Veriler, "Exp", 10);
var MA2 = Sistem.MA(Veriler, "Exp", 100);
Sistem.KesismeTara(MA1, MA2);
}
else
Sistem.Mesaj("Bu Sistemi Kullanmaya Yetkiniz Yoktur");
```

### - Liste Oluşturmak - Sistem.Liste(Deger)

IDEAL programında, Grafik Barlarına ait değerler, kapanışlar, indikatör değerleri, tüm hesaplatılan veya tanımlanan Çizgiler birer LİSTEDİR. Grafik üzerine bir ÇİZGİ çizmek için bir liste gerekir. Bu liste hazır fonksiyonlarla okutulan veya kullanıcı tarafından hesaplatılan veya sabit olarak tanımlanan bir liste olabilir. (Örneğin yatay biz çizgi). Bu fonksiyonun amacı, kodlama sırasında kullanılmak üzere çeşitli listeler tanımlamaktır.

### Fonksiyonun 2 Kullanım şekli vardır;

Sistem.Liste(x) = Bütün elemanlarının değeri x olan, üzerine uygulandığı grafikteki bar sayısı kaç ise, o kadar sayıda elemanı olan liste tanımlar;

Sistem.Liste(100,x) = Bütün elemanlarının değeri x olan ve eleman sayısı 100 olan liste tanımlar. Bu kullanım şekli, bir başka sembol veya periyoda ait veriler okutulunca ve oradan okunan veriler hesaplamalarda kullanılıp başka bir sembol veya periyotta kullanılacaksa tercih edilir.

iDeal sistem modülündeki veri veya fonksiyonların birçoğu LİSTE formatındadır. Örneğin *Sistem.GrafikFiyatSec("Kapanis*") fonksiyonu, bir grafikteki barların KAPANIŞ değerlerini içeren bir liste sunar. Her bir satırında bir SAYI olan bu listenin her bir elemanı grafikteki bir barı (tarihi) temsil eder.

### Listenin eleman sayısı = Bar Sayısıdır.

Bu listeyi kullanarak işlemler/hesaplamalar yapmak için bir değişkene atamak işi kolaylaştırır.

### ÖRNEK:

### var Close = Sistem.GrafikFiyatSec("Kapanis");

Bu satırdan sonra artık kapanış verilerini kullanmak istediğimiz yerde **"Close"** isimli listeyi/değişkeni kullanacağız.

Yeni ve/veya boş bir liste tanımlamak için Sistem.Liste(Deger) veya Sistem.Liste(BarSayisi, Deger) fonksiyonlarından biri kullanılır. (Robot kodu yazarken, listenin eleman sayısını da (bar sayısı) yazmak gerekir. Sistem yazarken bunu grafiğe uyguladığımız için, fonksiyon kaç adet bar olduğunu (listenin kaç elemanı olduğunu) kendisi bilir/bulur.

Sistem.Liste(0); //Bütün elemanları "0" olan (boş) bir listedir. (örneğin grafiğe 0 seviyesinde bir yatay çizgi olarak atanabilir)

Sistem.Liste(30); //Bütün elemanları "30" olan bir listedir. (örneğin grafiğe 30 seviyesinde bir yatay çizgi) Sistem.Liste(100,30); //100 adet elemanı olan ve her bir elemanı 30 olan bir listedir. Grafik üzerinden böyle bir liste çizdirilirse, çizgi ilk 100 barda gözükür.

Liste şeklinde hazır olarak elde edilen LİSTE'ler dışında, kullanıcı kendisi de LİSTE tanımlayabilir ve elemanlarını tek tek hesaplatıp oluşturabilir.

KURAL: İki liste arasında matematiksel işlem veya kıyasla yapılırken, bu işlemler TEK TEK LİSTENİN ELEMANLARI arasında yapılabilir. Yani Listere veri yazarken veya veri okurken LİSTENİN BİR ELEMANI (INDEXİ) belirtilmelidir. Indexleme işlemi için köşeli parantez kullanılır. Bir listenin x numaralı elemanı için ListeAdi[x] ifadesi kullanılır.

**ÖRNEK:** Listem isimli bir listenin 1000. Elemanının değerine ihtiyaç varsa; Listem[1000] şeklinde o elemana ulaşılır.

Bir listenin her elemanı okunacak veya yazılacaksa veya tek seferde birden fazla elemanına ihtiyaç varsa, liste üzerindeki okuma/yazma/hesaplama işlemleri BİR DÖNGÜ içinde yapılmalıdır. DÖNGÜLER belirtilen bir değerden (örnek ilk bardan) yine belirtilen bir son değere (son bara) kadar, her eleman için (her bar için) bir işlemi yapmak amacıyla kullanılır. (Bakınız Genel Kavramlar)

HATALI KULLANIM: Listem = (Listem1 + Listem2) / 2;

ÖRNEK: Bir sembolün her bir barının Yüksek ve Düşük değerlerinin toplamının yarısını bir başka listeye almak ve ekranda çizdirmek:

```
var Listem1 = Sistem.GrafikFiyatSec("Yuksek");
var Listem2 = Sistem.GrafikFiyatSec("Dusuk");
var Listem3 = Sistem.Liste(0); //Boş bir liste oluşturulur.
For (int i=1; i < Sistem.BarSayisi; i++)
{
Listem3[i] = Listem1[i] + Listem2[i]) / 2;
}
```

Yukarıdaki kodda, 1.bardan (i) BarSayısına kadar tim barlar için bir döngü kurulup, her bardayken, o bara karşılık gelen Listem1'in elemanı (Yuksek) ile Listem2'nin elemanı (Dusuk) toplanmış, çıkan sonuç 2'ye bölünmüş ve Listem3'ün o bara karşılık gelen elemanı bulunmuştur. Birkaç milisaniyede işletilen bu kod parçası sayesinde Listem3 oluşmuş olur. Yani grafiğin mesela 100. Barı için işlem şöyle olmuştur: Listem3[100] = (Listem1[100] + Listem2[100]) / 2

#### ÖRNEK: GRAFİK ÜZERİNDE PİVOTLARI HESAPLAMA GÖSTERME

```
var Veriler = Sistem.GrafikVerileri;
var sonbar = Veriler.Count-1;
var H = Veriler[sonbar-1].High;
var L = Veriler[sonbar-1].Low;
var C = Veriler[sonbar-1].Close;
var R = H - L;
var P = (H + L + C) / 3;
var R3 = Sistem.Liste(0);
var R1 = Sistem.Liste(0);
var R1 = Sistem.Liste(0);
var S1 = Sistem.Liste(0);
var S2 = Sistem.Liste(0);
var S3 = Sistem.Liste(0);
var pivot = Sistem.Liste(0);
```

for (int i=1; i<Sistem.BarSayisi; i++)</pre> R3[i] = P + (R \* 1.0f);R2[i] = P + (R \* 0.618f);R1[i] = P + (R \* 0.382f);S1[i] = P - (R \* 0.382f);S2[i] = P - (R \* 0.618f); S3[i] = P - (R \* 1.0f); pivot[i] = P; } Sistem.Cizgiler[0].Deger=R3; Sistem.Cizgiler[1].Deger=R2; Sistem.Cizgiler[2].Deger=R1; Sistem.Cizgiler[3].Deger=S1; Sistem.Cizgiler[4].Deger=S2; Sistem.Cizgiler[5].Deger=S3; Sistem.Cizgiler[6].Deger=pivot;

- LLV - Sistem.LLV(14)

LLV (Lowest Low Value) isimli indikatörü hesaplayıp çağıran ideal sistem fonksiyonudur. 1 adet parametre alır ve varsayılan değeri 14'tür. LLV (Lowest Low Value) girilen parametre kadar bar için görülen EN DÜŞÜK değeri bir indikatör olarak hesaplar çizer. Grafik barları üzerine çizdirilir ve böylece fiyat kesişimleri veya kanal analizi yapılabilir. Kullanım ve yazım şekilleri aşağıdaki gibidir;

Sistem.LLV(14); Sistem.LLV(14, Liste); Sistem.LLV(14, "Dusuk"); Sistem.HHV(GrafikVerileri, 14, "Dusuk"); //Başka bir sembole veya grafik periyoduna ait barlar için LLV hesaplar.

Örnek: İndikatörü çağırıp çizdirmek

var LL = Sistem.LLV(14); Sistem.Cizgiler[0].Deger = LL;

- Lot - Sistem.Lot(Sembol)

Bir sembolün o an itibariyle ve çeşitli zaman dilimleri için kaç lot işlem gördüğü bilgisini okumak için kullanılır. Fonksiyonun içine Lotu okutulmak istenen sembol yazılır.

<u>Not:</u> iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir. Örneğin USDTYR için FX'USDTRY şeklinde yazılır. Bir sembolün PİYASA kodunun ne olduğu, o sembolü sayfanıza yazarken @ işareti yanında gösterilir.)

Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.LotAltiAy(Sembol);
Sistem.LotBirAy(Sembol);
Sistem.LotBirHafta(Sembol);
Sistem.LotBirYil(Sembol);
Sistem.LotBuAy(Sembol);
Sistem.LotBuHafta(Sembol);
```



Sistem.LotBuYil(Sembol);
Sistem.LotGun(Sembol);
Sistem.LotUcAy(Sembol);

Örnek: hisse senedinin, bu doküman yazılırken ki son 6 ayın en düşük değerinin okuyup ekrana mesaj olarak çıkaran kod:

```
var AltiAyLot = Sistem.LotAltiAy(Sembol);
Sistem.Mesaj("6 Aylık işlem hacmi lot bazında = " + AltiAyLot.ToString());
```

### - LY - Sistem. LY(Periyot)

Bir grafik üzerinde, girili periyot içerisinde görülen EN DÜŞÜK değerden yüzdesel olarak ne kadar yuzakta olduğumuz bilgisini dönen bir indikatör olan LY indikatörünü hesaplar. Bir adet parametre alır. Kullanım şekli Sistem.LY(20); şeklindedir.

Örnek: Momentum kendi ortalamasının altındaysa, fiyatın 20 barlık en düşüğünden uzaklığı %2 den küçükse AL, tersi durumda SAT örnek sistemi.

```
var LY = Sistem.LY(20); // 20 periyottaki en düşükle fiyat arasındaki yüzde fark
var Mom = Sistem.Momentum(5);
var AvrMom = Sistem.MA(Mom, "Simple", 10);
var SonYon = "";
for (int i=1; i < Sistem.BarSayisi; i++)</pre>
{
   LY[i] = Math.Abs(LY[i]);
   if (SonYon != "A" && LY[i] < 3 && Mom[i] > AvrMom[i])
   {
      SonYon = "A";
      Sistem.Yon[i] = "A";
   }
   if (SonYon != "S" && LY[i] > 3 && Mom[i] < AvrMom[i])
   {
      SonYon = "S";
      Sistem.Yon[i] = "S";
   }
}
```

- MA – Moving Average (Hareketli Ortalamalar) - Sistem. MA(....)

Bir grafik için hareketli ortalamaları hesaplatmak/çizdirmek için iDeal Sistem kütüphanesinde Sistem.MA fonksiyonları bulunmaktadır. MA (Moving Average) indikatörü hesaplamak için 3 parametreyi girdi olarak vermek gerekir;

- Periyot (kaç barlık ortalama alınsın?)
- Yöntem (Hareketli ortalam hesaplama yöntemi ne olsun? (basit, üssel, ağırlılı vs.))
- Veri (hangi verilerin ortalaması alınsın? (kapanışların, yükseklerin, hacimin, bir başka indikatörün vs.)

Aşağıda gösterildiği şekilde 2 kullanım şekli vardır;

```
Sistem.MA(10, "Simple", "Kapanis");
Sistem.MA(Liste, "Exp", 10);
```



Birinci kullanım şekli, sadece yazılan kdoun grafiğe uygulandığı durumlarda kullanılır ve üzerine uygulandığı grafiğin barlarının KAPANIŞ değerlerinin üssel 10 perityotluk ortalamasını hesaplar.

İkinci kullanım şekli, MA'sı hesaplanmak istenen data eğer bar kapanışları eğer grafiğin barlarına ait yüksek, düşük, açılış, kapanış vs. değil de başka bir liste veya indikatör ise tercih edilir. Örneğin RSI indikatörümnün ortalaması hesaplatılacaksa bu kullanım şekli yazılır ve Liste yazan yere RSI için kullanuılan isim yazılır.

Barlara Ait Veri Setleri:

Bir grafik penceresinde ekran görülen her bir bar/mum aşağıdaki bilgileri içerir ve bu verilerin ortalamaları hesaplatılmak istenebilir. Bu durum durumda Sistem.MA fonksiyonunda çift tırnak için neyin yazılması gerektiğini, kod yazdığınız alanın sağ üst tarafında yer alan GRAFİK VERİLERİ isimli kutudan görebilirsiniz.

Grafik Verileri	Kapanis 🔷 👻
	Acilis
	Dusuk
	Hacim
	Kapanis
	Lot
	OHLC/4
	Ortalama
	OrtaNokta
	Tipik
	Yuksek

Aynı şekilde MA hesaplama yöntemlerinin neler olduğu ve nasıl yazılacağını da ORTALAMA isimli listeden görebilirsiniz.



Hareketli ortalama yöntemleri hakkında kısa açıklama notları da ayrıca aşağıda belirtişlmiştir;

- Simple-(Basit): Verilerin hepsi eşit ağırlığa sahip olarak kullanılır. Mesela son 5 günün kapanış ortalamasında, 5 kapanışın toplamı 5'e bölünerek bulunur.
- Exp-(Üssel): Son gün en ağırlıklı olmak üzere, son günden geriye doğru ortalama ağırlıkları üssel olarak hesaplanır. En çok kullanılan yöntemdir.

SEZAİ KILIÇ

- Weighted(Ağırlıklı): Ağırlık son günlere kaydırılır. Mesela 1. gün 1 ile, 2. gün 2 ile, .....5. gün 5 ile çarpılır. Toplam 15 e (1+2+3+4+5) bölünür.
- Triangle-(Üçgensel): Belirlenen periyodun orta kısmına düşen günlere daha fazla ağırlık vererek hesaplanan bir hareketli ortalamadır.
- Variable-(Değişken) : Bu da bir çeşit üssel hareketli ortalamadır.
- Zero Lag: Fiyat ile ortalama arasındaki açıklığı olabildiğince daraltmayı hedefleyen bir indikatördür. Fiyat hareketlerine daha duyarlı bir grafik verir.
- Welles Wilder: Bu ortalama şeklini türeten analistin adı ile anılmaktadır. Ağırlıklı ortalamaya benzer. Eski fiyatlar ağırlıklı ortalamaya daha az katkı sağlarken, yakın tarihli fiyatlar, ortalama üzerinde daha fazla ağırlık sahibidir
- HullMA: Bu ortalama bir listenin Hull ortalamasını alı.

Örnek: DI+ indikatörününü ortalamasını alıp çizdirmek;

```
var A = Sistem.DirectionalIndicatorPlus(14);
var B = Sistem.MA(A, "Exp", 10);
Sistem.Cizgiler[0].Deger = A;
Sistem.Cizgiler[1].Deger = B;
```

#### MACD - Sistem. MACD(12,26)

MACD indikatörünü çağırır. 2 Adet parametre alır (varsayılan 12 ve 26) . Aşağıdaki gibi 3 yazım şekli vardır;

```
Sistem.MACD(12,26);
Sistem.MACD(Liste,12, 26);
Sistem.MACD(Veriler, 12, 26);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var MACD = Sistem.MACD(12,26);
var AVR = Sistem.MA(MACD,"Simple", 9);
Sistem.Cizgiler[0].Deger = MACD;
Sistem.Cizgiler[1].Deger = AVR;
```

- MAFARK - Sistem. MAFARK(Liste, "Exp", 9)

Bir fiyat veya indikatörün kendi ortalamasıyla arasındaki farkı hesaplayın getiren iDeal Sistem fonksiyonudur.Kullanım şekli aşağıdaki gibidir;

Sistem.MAFARK(Liste, "Simple", 10);

<u>Örnek: Momentum İndikatörünün kendi 10'luk basit ortalamasıyla arasındaki farkı</u> hesaplayıp bir indikatör gibi çizmek

```
var MOM = Sistem.Momentum(12);
var FARK = Sistem.MAFARK(MOM, "Simple", 10);
```

```
Sistem.Cizgiler[0].Deger = FARK;
```

# ideal

### - MAM (MA'ların MA larını almak) - Sistem. MAM(Liste, "Exp", P1, P2, P3, P4....)

MA (Moving Average) bir fiyat veya indikatörün hareketli ortalamasını almak için kullanılan sistem fonksiyonudurç MAM ise aynı ortalama alma işini üst üste birden fazla kez yapmak istendiğinde kullanılır. Örneğin fiyatın (kapanışın) MA'sını hesaplatıp, hesaplanan MA'nın tekrar MA'sını almak, sonra bunun da MA'sını almak ve böyle defalarca kez (genelde fibonacci sayıları ile artarak giden MA'lar almak) için kullanılır. Kullanım şekli aşağıdaki gibidir;

Sistem.MAM(Liste, "Weighted", 3, 5, 8, 13, 21, 34, 55, 89);

Örnek: Sosyal paylaşım sitelerinde Mavilim ismiyle paylaşılan açık kaynak kodlu indikatör, kapanışların üst üste 6 kez MA'sının alınmış halidir ve aşağıdaki örnekte MAM fonksiyonu ile tek satırda hesaplatılabilir.

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var Mavilim = Sistem.MAM(C, "Weighted", 3, 5, 8, 13, 21, 34);
Sistem.Cizgiler[0].Deger = Mavilim;
```

### - MAIL GÖNDERMEK - Sistem. MailGonder.....(...)

iDeal Sistem kütüphanesinde yer alan Mail Gönder fonksiyonu ile ister belli zaman aralıklarında, ister belli koşullar gerçekleştiği zaman, ister robotlarını al/sat yaptığında E-Mail gönderimi yapmak mümkündür. Bu amaçla Sistem.MailGonder fonksiyonu kullanılır.

Mail Gönder fonksiyonu, beraberinde sunulan alt fonksiyon sayesinde, maile dosya ekleme imkânı da sunar.

Siz bilgisayar başında olmasanız bile, istediğiniz zaman istediğiniz bilgi, belge ve görüntüleri otomatik olarak (Robot üzerinden) mail attırtabilirsiniz.

Çok çeşitli amaçlarla E-Mail gönderme özelliğini kullanabilirsiniz.

- Seyahatte, toplantıda, tatilde olduğunuz zaman, bilgisayarınızın durumu, piyasalardan anlık bilgiler, belirlediğiniz kriterlerinin gerçekleştiğine dair alarm gibi amaçlarla kendinize mail attırtıp, telefondan her şeyi kontrol altında tutabilirsiniz.
- Ekranda hazırladığınız teknik analiz veya veri şablonlarını, belli aralıklar veya koşullara bağlı olarak, fotosunu çektirtip, abone, personel, takipçi veya arkadaşlarınıza göndertebilirsiniz.
- Otomatik AL/SAT yapan robotunuz varsa ve bir sanal/kiralık sunucu üzerinde çalışıyorsa, işlem yaptıkça, sinyal geldikçe size mail ile bilgi vermesini sağlayabilirsiniz

**Sistem.MailGonder()** fonksiyonu, beraberinde sunulan bazı alt fonksiyonlarla birlikte kullanılır.

Kodun sizin adınıza mail atabilmesi için, maili göndereceği adresi ve bu adres ait şifreyi, IP ve Port bilgilerini bilmesi gerekir. Hotmail, Gmail, Yandex, Yahoo gibi ücretsiz mail sağlayıcıları için bu bilgiler yaygın olarak bilir veya internetten kolayca öğrenilir. Kurumsal bir mailiniz varsa

veya başka bir sağlayıcıdan bir E-Mail adresiniz varsa IP/Port bilgilerini teknik birimden öğrenebilirsiniz.

Mail Gönder fonksiyonuyla birlikte kullanılabilecek Alt Fonksiyonların listesi aşağıdadır:

- Sistem.MailServerAdres = "smtp.gmail.com"; //Çift tırnak içinde sizin mail server IP/Host bilginiz.
- **Sistem.MailServerPort** = 587; //Gmail'in mailserver portu.
- Sistem.MailKonu = "Robotum bilgilendirme";
- **Sistem.MailMetin** = "Buraya ister yazı ister koddan veri alınabilir";
- Sistem.MailGonderenAdres = "abc123@gmail.com"; //Maili gönderen kişinin mail adresi
- Sistem.MailGonderenSifre = "xxxxxxx"; //Göndericinin mail şifresi
- Sistem.MailAliciEkle(<a href="mailto:sezai.kilic@directfn.com">sezai.kilic@directfn.com</a>, deneme12@yahoo.com</a>); //Mailin alıcıları
- Sistem.MailGonder();

**NOT-1:** Alıcı satırına, çift tırnak içinde, aralarına virgül konulara istenildiği kadar alıcı mail adresi eklenebilir.

**NOT-2:** CC ve BCC yapmak için de iki alt fonksiyon vardır. Atılan maili bu fonksiyonları kullanarak bilgi veya gizli alıcı da ekleyebilirsiniz.

- Sistem.MailBccEkle("adres");
- Sistem.MailCcEkle("adres");

**NOT-3:** Robotunuzun, Mail ekine bilgisayarınızda bulunan bir dosyayı eklemesi için yukardaki bloğun içine (MailGonder satırından önceye) aşağıdaki satırı ekleyebilirsiniz:

Sistem.MailDosyaEkle("C:\\test.png");

**NOT-4:** Çok popüler Ücretsiz Mail adresi sağlayıcılarının MailServer adresleri ve Port numaraları aşağıda belirtilmiştir. (başka mail servis sağlayıcılar veya aşağıdaki sağlayıcılar için güncel adres/port bilgilerini internetten aratarak kolayca bulabilirsiniz)

- Yahoo smtp.mail.yahoo.com 587
- o Gmail smtp.gmail.com 587
- Hotmail smtp.live.com 587
- Yandex smtp.yandex.com 587

**NOT-5:** Mail gönder fonksiyonu çalıştığı zaman, ekrana mail gönderildi şeklinde bir mesaj çıkarılır. Başka birine şifreli formül sağlayıp, formülü verdiği kişinin haberi olmaksızın mail gönder fonksiyonu kullanımı denenmesin veya denenirse de bunu kullanan kişi farkında olsun diye böyle bir zorunlu mesaj koda gömülmüştür

ÖRNEK: GARAN Hisse fiyatı 7.50 olursa ekran görüntüsünü de çekip ekleyrek mail gönder:

```
var Sembol = "IMKBH'GARAN";
var Seviye = 7.50;
var SonFiyat = Sistem.SonFiyat(Sembol);
```

```
var Mesaj = Sembol + "Son fiyatı " + SonFiyat + " değerini kırdı";
```

```
if (SonFiyat >= Seviye)
{
    Sistem.GoruntuKaydet("C:\\Ekranım.png");
    var MailServer = "smtp.gmail.com";
    Sistem.MailServerAdres = MailServer;
    Sistem.MailServerPort = 587;
    Sistem.MailServerPort = 587;
    Sistem.MailMetin = Mesaj;
    Sistem.MailGonderenAdres = "deneme123@gmail.com";
    Sistem.MailGonderenSifre = "xxxxxxx";
    Sistem.MailDosyaEkle("C:\\Ekranım.png");
    Sistem.MailAliciEkle("sezaik@idealdata.com.tr");
}
```

### - MASS INDEX - Sistem.MassIndex(25)

Mass Index olarak bilinen indikatörü çağırır. 1 Adet parametre alır (varsayılan 25) . Aşağıdaki gibi 2 yazım şekli vardır;

```
Sistem.MassIndex(25);
Sistem.MassIndex(Veriler, 25);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var MASS = Sistem.MassIndex(25);
Sistem.Cizgiler[0].Deger = MASS;
```

- MEDIAN - Sistem.Median(Liste,14) ve Sistem.MedianDeger(Liste)

Bir listenin (fiyat veya indikatör) Median değerini (medyan ortalamasını) hesaplatmak için iDeal Sistem Kütüphanesinde 2 adet fonksiyon yer almaktadır. Bunlardan biri belli bir zaman veya periyot dilimi için sürekli median hesaplayıp bir indikatör elde ederken, diğeri ise tüm data listesinin median ortalamasının sayıdal değerinin kaç olduğunu hesaplar.

Kullanım şekillleri aşağıdaki gibidir;

```
Sistem.Median(Liste, 14); //hep son 14 barın median ortalamasını hesaplar ve bir indikatör
elde eder
Sistem.MedianDeger(Liste); //Data olarak verilmiş listenin tüm elemanlarının median
ortalamasının kaç olduğunu hesaplar
```

Örnek: Kapanışların 14'lük Median ortalamasını indikatör olarak çizdirmek ve tüm tarihsel kapanış verilerinin Median ortalamasını ekrana mesaj olarak bastırmak.

```
var TimePeriod = 14;
var Data = Sistem.GrafikFiyatSec("Kapanis");
var m = Sistem.Median(Data, TimePeriod);
var m2 = Sistem.MedianDeger(Data);
```


Sistem.Cizgiler[0].Deger = m; Sistem.Mesaj(m2.ToString());



### - MESAJ - Sistem.Mesaj(Metin)

İstenilen bir anda veya bir koşula bağlı olarak, ekrana bir mesaj çıkartmak için kullanılır. Mesaj fonksiyonu bazen bir alarm gibi (bir koşul gerçekleştiği anda haberdar olmak için) kullanılabileceği gibi bazen de yazılan kodda tanımlanan/hesaplanan bir değişkenin veya listenin boyutu, değeri vs var mı, doğru mu gibi kontroller (debug) için kullanılır.

Mesaj fonksiyonu, ekrana açılan bir kutu içinde, bir METNİN yazdırılması işini yapar. Ekrana açılan pencerede gösterilecek bilgi bir fiyat (sayı), zaman (tarih/saat) veya bir cümle de olsa, tüm ifadeler bir TEXT (METİN) olarak karşımıza gelir.

Programlama dillerinde METIN(TEXT/YAZI) ifadeleri tipine STRING denir.

Bu nedenle ekrana bastırılacak sonuç, eğer kodlama "string" tipinde (bakınız değişken tipleri) tanımlanmamışsa, mesaj penceresine aktarılırken sonuna STRINGE ÇEVİR anlamına gelen

".ToString() ibaresi eklenir. (Tüm C# Komutlarında olduğu gibi kelimelerin ilk harfleri büyük yazılmalıdır.)

4 ayrı mesaj fonksiyonu imkanı vardır. Ekrana açılacak mesaj pencerelerinin genişliği, yüksekliği, ekranın en solundan kaç pixel uzakta ve ekranın en üstünden kaç pixel uzakta açılması istendiği, içindeki yazının rengi kullanıcılar tarafından (istenirse) girilebilir. Tüm kullanım şekilleri aşağıdaki gibidir;

Sistem.Mesaj(metin); Sistem.Mesaj(metin, renk); Sistem.Mesaj(metin, renk, sol, ust, genislik, yukseklik); Sistem.Mesaj(metin, sol, ust, genislik, yukseklik); Burada Renk bilgisi Color.Blue seklinde (İngilizce renk ismi) ve boyut/konum verisi de tam sayı (50, 180, 500 vs.9 olarak girilir. string strmsg = "iDeal Dünya"; Sistem.Mesaj(strmsg); var SonFiyat = Sistem.SonFiyat("IMKBH'GARAN"); Sistem.Mesaj(SonFiyat.ToString()); var zaman = Sistem.Tarih; Sistem.Mesaj(zaman.ToString()); 🖳 iDeal Mesaj 08.10.2015 16:47:47 2015.10.08 iDeal Mesaj 08.10.2015 16:48:16 7.71 iDeal Mesaj 08.10.2015 16:48:45 iDeal Dünya

- Momentum - Sistem.Momentum(12)

Momentum olarak bilinen indikatörü çağırır. 1 Adet parametre alır (varsayılan 12) . Aşağıdaki gibi 2 yazım şekli vardır;

```
Sistem. Momentum(12);
Sistem. Momentum(Liste,12);
Sistem. Momentum(Veriler,12);
```

Örnek: İndikatörü ve ortalamasını çağırıp çizdirmek

```
var MOM = Sistem.Momentum(100);
var ORT = Sistem.MA(MOM, "Exp", 10);
Sistem.Cizgiler[0].Deger = MOM;
Sistem.Cizgiler[1].Deger = ORT;
```

### - Money Flow Index (Para Akış Endeksi) - Sistem.MoneyFlowIndex(14)

MFI veya Money Flow Index veya Para Akış Indeksi olarak bilinen indikatörü çağırır. 1 Adet parametre alır (varsayılan 14). Aşağıdaki gibi 2 yazım şekli vardır;

<u>NOT:</u> Bu indikatörün anlık olarak hesaplanması için Anlık olarak hacim bilgisi görme yetkisi gerekir. Bu da Borsa İstanbul Piyasaları için Düzey1 Plus (1 kademe derinlik) olarak bilinen yetkidir.

```
Sistem. MoneyFlowIndex(14);
Sistem. MoneyFlowIndex (Veriler,14);
```

Örnek: İndikatörü ve ortalamasını çağırıp çizdirmek

```
var MFI = Sistem.MoneyFlowIndex(14);
var ORT = Sistem.MA(MFI, "Exp", 10);
Sistem.Cizgiler[0].Deger = MFI;
Sistem.Cizgiler[1].Deger = ORT;
```

- <u>Multi Sembol Birleştir Aynı Yön</u> <u>Sistem.MultiSembolBirlestirAyniYon(Sistem,Sembol1, Sembol2)</u>

Al ve Sat sinyalleri üreten bir sistemin, birden fazla sembolün grafiğinde uygulanması ve hepsinde birden aynı yönde olunduğu zaman o yönde sinyal üreten yeni bir sistem elde edilmesi amacıyla kullanılan bir iDeal Sistem Fonksiyonudur.

Aşağıdaki şekilde kullanılır;

Sistem.MultiSembolBirlestirAyniYon("referanssistem","IMKBX'XU030","IMKBX'XU100","IMKBX'XBANK");

Bu komutu yazıp kaydeden bir ideal kullanıcısı, adı ReferansSistem olan formülünün XU030, XU100 ve XBANK grafiklerindeki yönü hepsinde birden AL olunca AL sinyali üretir, hepsinde bir SAT yönünde olduğu barda ise SAT sinyali üretir. Bu şekilde yazılıp kaydedilen bu tek satırlık formül istenilen herhangi bir senet veya vadelide çalıştırılabilir. Formül hangi zaman periyotlu grafiğie uygulanırsa, diğer sembollerdeki yön bilgisi de o sembollerin grafiklerinin aynı zaman periyotlu grafiklerinden elde edilir.

ÖRNEK: MA2 isimli sistem xu030/xu100/xbank grafiklerinde aynı abrda al olduğu anda VİOP Endesks grafiğinde AL verir (veya sat)



- NVI – Negative Volume Index - Sistem.NegativeVolumeIndex()

NVI veya Negative Volume Index olarak bilinen indikatörü çağırır. İndikatör parametre almaz. Aşağıdaki gibi 2 yazım şekli vardır;

<u>NOT:</u> Bu indikatörün anlık olarak hesaplanması için Anlık olarak hacim bilgisi görme yetkisi gerekir. Bu da Borsa İstanbul Piyasaları için Düzey1 Plus (1 kademe derinlik) olarak bilinen yetkidir.

Sistem. NegativeVolumeIndex();
Sistem. NegativeVolumeIndex(Veriler);

Örnek: İndikatörü ve ortalamasını çağırıp çizdirmek

```
var NVI = Sistem.NegativeVolumeIndex();
var ORT = Sistem.MA(MFI, "Exp", 10);
Sistem.Cizgiler[0].Deger = NVI;
Sistem.Cizgiler[1].Deger = ORT;
```

- NESNE Kullanımı (Nesne Getir/ Kaydet) - Sistem.NesneGetir()

Formül yazarken, bazı sayı, metin veya listeleri değişken tanımlayıp (var x....) değişkenlerde saklar kod yazanlar. Fakat kod içinde tanımlanmış değişkenler, kod baştan sona her seferinde yeniden çalıştığında, tanımladıkları değişkenler bir kez ilk tanımlandıkları zaman atanan değerlerine döner ve kodun devamında yeniden hesaplamar yapılarak güncellenirler. Oysa bazı durumlarda, çeşitli bilgilerin tutulduğu değişkenlerdeki bilgilerin hiç değişmemesi veya sadece gerektiğinde değişmesi gerekir. Bu gibi durumlarda iDeal kullanıcıları Sistem.SayiTablosu, Sistem.SozcukTablosu gibi bilgisayar belleğinde duracak şekilde bilgi kaydetmesi ve gerektiğinde oradan okumasını imkanlarını sunan fonksiyonları kullanabilir. Sayı ve Sözcük tablolarında isimlerinde belirtilen gibi veriler saklanabilirken, NesneKaydet fonksiyonu ile bellekte bir obje, bir sözlük, bir liste, bir dizi veya tarih/saat bilgisi saklanabilir, kaydedilebilir ve gerektiğinde kullanılabilir.

NesneKaydet fonksiyonu, kullanıcı istediğinde bir veriyi kaydetmek/saklamak için, NesneGetir fonksiyonu ise gerektiğinde buradaki bilgiyi çağırıp okumak için kullanılır. Hiç veri saklanmamışken NesneOku denildiğinde NULL döner.

Aşağıda nesne kullanımı için 2 örnek gösterilmiştir.

ÖRNEK-1: Bu örnekte, son barın tarihi nesneye kaydedilmiş ve her seferinde en son barın tarihi nesneye yazılmış tarihle kıyaslanmış, bu sayede grafiğe yeni bir bar geldiği tespit edilmiştir.

```
var V = Sistem.GrafikVerileri;
var Anahtar = Sistem.Name + ";" + Sistem.Sembol + ";" + Sistem.Periyot;
var SonBarTarih = Sistem.NesneGetir(Anahtar);
if (SonBarTarih == null)
{
    SonBarTarih = V[V.Count - 1].Date;
    Sistem.NesneKaydet(Anahtar, SonBarTarih);
}
if (V[V.Count - 1].Date > SonBarTarih)
{
    Sistem.Mesaj(V[V.Count - 1].Date.ToString("HH:mm"));
    SonBarTarih = V[V.Count - 1].Date;
    Sistem.NesneKaydet(Anahtar, SonBarTarih);
}
```

ÖRNEK-2: Bir Excel dosyasına, fiyatı bizim belirlediğimiz seviyeye düşünce, bizim belirkediğimiz adet alım yapan, yine bizim belirlediğimiz KAR AL ve STOP fiyatlarına ulaşınca da hisseleri satan bir robot düşünelim. Bu robot, hisseleri, alım seviyelerini, kar ve stop seviyelerini, işlem adetlerini EXCEL dosyasından okusun. Okunan Excel dosyası Nesneye kaydedilip, robot tarafından kullanılsın.

	А	В	С	D	E	
1	SENET	AL SEVİYESİ	KAR AL	STOP	Miktar	
2	GARAN	6.95	7.35	6.70	1000	
3	EREGL	9.20	10.30	8.90	500	
4	DOHOL	1.20	1.35	1.05	1500	
5	FENER	29.50	35.00	27.50	100	
6	KRDMD	3.00	3.40	2.90	500	
7	GSRAY	4.90	5.80	4.75	5000	

### Not1: C:\ideal\HisseRobot.xlsx isimli bir excel dosyası kullandık Not:2 Excel satır ve sütunlarında örnek olarak aşağıdaki bilgileri yazılı idi.

### **ROBOT KODU:**

string FileName = "C:\\ideal\\HisseRobot.XlsX"; bool DevamEt = true;

if (DateTime.Now.DayOfWeek == DayOfWeek.Saturday) DevamEt = false;

```
if (DateTime.Now.DayOfWeek == DayOfWeek.Sunday) DevamEt = false;
if (DateTime.Now.ToString("HHmm").CompareTo("1000") <= 0) DevamEt = false;
if (DateTime.Now.ToString("HHmm").CompareTo("1759") >= 0) DevamEt = false;
if (System.IO.File.Exists(FileName) == false) DevamEt = false;
if (DevamEt)
{
    var ExcelArray = Sistem.NesneGetir(FileName + ";" + DateTime.Now.ToString("yyyyMMdd"));
    if (ExcelArray == null)
    {
         ExcelArray = Sistem.ExcelOku(FileName);
         Sistem.NesneKaydet(FileName + ";" + DateTime.Now.ToString("yyyyMMdd"), ExcelArray);
    int SatirSayisi = ExcelArray.GetLength(0); //satirlarin sayisini bul
    for (int i = 2; i <= SatirSayisi; i++)</pre>
    {
         var Sembol = ExcelArray[i, 1].ToString();
         var AlisFiyat = (decimal)ExcelArray[i, 2];
         var HedefFiyat = (decimal)ExcelArray[i, 3];
         var StopFiyat = (decimal)ExcelArray[i, 4];
        var Lot = (int)ExcelArray[i, 5];
var Anahtar = Sistem.Name + " , " + Sembol;
         double IslemFiyat = 0;
         DateTime IslemTarih;
         var Rezerv = "";
         var Pozisyon = Sistem.PozisyonKontrolOku(Anahtar, out IslemFiyat, out IslemTarih, out
Rezerv);
         var EmirSembol = "IMKBH'" + Sembol;
         var basicitem = Sistem.YuzeyselVeriOku(EmirSembol);
         var sonfiyat = (decimal)basicitem.LastPrice;
         var bidfiyat = (decimal)basicitem.BidPriceDec;
         var askfiyat = (decimal)basicitem.AskPriceDec;
         if (sonfiyat == 0) continue;
         if (bidfivat == 0) continue;
         if (askfiyat == 0) continue;
         var Islem = "";
         var Miktar = 0.0;
         if (sonfiyat < AlisFiyat && Pozisyon == 0 && Rezerv == "") // AL
         {
            Rezerv = "AL";
            Miktar = Lot;
            IslemFiyat = Sistem.SonFiyat(EmirSembol);
         else if (Pozisyon > 0 && askfiyat >= HedefFiyat && Rezerv == "AL") // KARLA KAPAT
         {
            Rezerv = "KAR AL";
            Miktar = -Lot;
         }
         else if (Pozisyon > 0 && bidfiyat < StopFiyat && Rezerv == "AL") // STOP
         {
            Rezerv = "STOP";
            Miktar = -Lot;
         }
         if (Miktar > 0) Islem = "ALIS";
         if (Miktar < 0) Islem = "SATIS";</pre>
         if (Islem != "")
         {
            Sistem.PozisyonKontrolGuncelle(Anahtar, Miktar + Pozisyon, IslemFiyat, Rezerv);
            Sistem.EmirSembol = EmirSembol;
            Sistem.EmirIslem = Islem;
            Sistem.EmirSuresi = "KIE"
            Sistem.EmirTipi = "Piyasa";
            Sistem.EmirMiktari = Math.Abs(Miktar);
            Sistem.EmirGonder();
        }
   }
```

#### }

ÖRNEK3: Robot işlem yaptıktan sonra 20 saniye boyunca dursun, sonra devam etsin;

```
//i$LEM YAPTIKTAN SONRA ROBOT 20 SANİYE DURSUN
//SONRA DVAM ETSİN
var DevamEt = false;
var Anahtar = Sistem.Name + ", ZAMAN KONTROL";
var Saat = Sistem.NesneGetir(Anahtar);
if (Saat == null)
    DevamEt = true;
else if (DateTime.Now > Saat.AddSeconds(20))
    DevamEt = true;
if (DevamEt)
{
    var Islem = Sistem.RobotViopAktif("Sistem1", "VIP'VIP-X030", Sistem.AktifViopKontrat, "5", 4);
    if (Islem != "")
        Sistem.NesneKaydet(Anahtar, DateTime.Now);
}
```

### - NET HACIM Okuma Fonksiyonları – Sistem.NetHacim(...)

iDeal Sistem kütüphanesinde Net Hacim olarak ekranlarda izlediğimiz verilere erişim için tanımlanmış 4 adet fonksiyon bulunmaktadır. Fonksiyonların kullanım şekilleri ve sunduğu bilgilerin neler olduğu aşağıdadki gibidir; (Not: en az düzey1 plus Borsa lisansı gerekir)

### • Sistem.NetHacim(Sembol):

Bu fonksiyon, herhangi bir hisse veya vadeli kontratın, bulunduğumuz gün, bu komutu çalıştırdığımız an itibariyle TL bazlı NET hacim bilgisini (yani alıştan olan işlemlerin hacmi ile satıştan olan işlemlerin hacmi arasındaki farkını) döner.

### Örnek: EKGYO hissenin şu an itibariyle net hacmi kaç TL?

```
var Sembol = "IMKBH'EKGYO";
var NetHacim = Sistem.NetHacim(Sembol);
Sistem.Mesaj(NetHacim.ToString("0,000"));
var Sembol = "IMKBH'EKGYO";
var NetHacim = Sistem.NetHacim(Sembol);
```

```
Sistem.Mesaj(NetHacim.ToString("0,000"));
```

SplitKullanim

**|**17,178,252

## • Sistem.NetHacimOran(Sembol):

Bu fonksiyon, herhangi bir hisse veya vadeli kontratın, bulunduğumuz gün, bu komutu çalıştırdığımız an itibariyle TL bazlı NET hacim bilgisinin (yani alıştan olan işlemlerin hacmi arasındaki farkının) o hissede o gün o ana kadarki toplam hacime olan oranını döner.

### Örnek: GARAN hissenin şu an itibariyle Net Hacim / Tooplam Hacim oranı nedir?

```
var Sembol = "IMKBH'GARAN";
var NetOran = Sistem.NetHacimOran(Sembol);
Sistem.Mesaj(NetOran.ToString("0.00"));
```

- Sistem.NetHacimBist():
- Sistem.NetHacimBist():

Bu 2 fonksiyon ise, tüm BIST pay piyasasının Net Hacim bilgisi ile NetHacim/ToplamHacim oranı bilgilerini elde etmek için kullanılır.

BIST menüsü altından açılabilen TÜM İŞLEMLER isimli pencerenin üst kısmında, her bir işlem oldukçşa anlık olarak hesaplanıp gösterilen aşağıdaki görseldeki bilgilere erişim için bu fonksiyonlar kullanılır.

🛎 🗧 Bist	Tüm Semb	oller Bug	ün Ack	Haci n ≥ 0	Filtre		_
Hacim 27,415,73	Para 9,608 15,0	a+ 163,953,079	Para- 12,351,78	Net 86,52 2,71	2,166,550	Net% 9.89	
No	İşlem	Saat	Senet	Fiyat	Lot	Alan	Satan
2683535	A2DA0C10	17:11:33	FENER	29.82	160	IS Yat	Tacirler
2683534	A2DA0C10	17:11:33	BRISA	11.51	503	Tacirler	Uniu Menkul
2683533	A2DA0C10	17:11:33	BRISA	11.51	5	Tacirler	Oyak
2683532	A2DA0C10	17:11:33	BRISA	11.51	100	Tacirler	Seker Yat.
2683531	A2DA0C20	17:11:33	ULAS	2.96	50	Deniz	Gedik
2683530	A2DA0C10	17:11:33	ADESE	2.38	2	A1 Capital	Oyak
2683529	A2DA0C10	17:11:33	ESCOM	4.09	778	Tacirler	Zíraat Yat.
2683528	A2DA0C10	17:11:33	ESCOM	4.09	222	Tacirler	Osmanli
2683527	A2DA0C20	17:11:33	TUPRS	79.85	15	IS Yat	Garanti Yat.
2683526	A2DA0C20	17:11:33	ULUUN	10.90	950	Info	Global
2683525	A2DA0C20	17:11:33	VAKKO	6.27	5	Garanti Yat.	YapiKredi Yat.
2683524	A2DA0C20	17:11:33	THYAO	10.53	2407	Alnus	Garanti Yat.
2683523	A2DA0C20	17:11:33	THYAO	10.53	593	Meksa	Garanti Yat.
2683522	A2DA0C20	17:11:33	PENGD	3.59	1396	Bizim	Vakif
2683521	A2DA0C10	17:11:33	AKBNK	5.00	- 4	Tacirler	Oyak
2683520	A2DA0C20	17:11:33	MARTI	1.21	50	Alnus	Merrill Lynch
2683519	A2DA0C20	17:11:33	TATGD	10.46	3000	YapiKredi Ya <u>t.</u>	Meksa
2683518	A2DA0C10	17:11:33	GARAN	7.02	81	IS Yat	Garanti Yat.
2683517	A2DA0C10	17:11:33	GARAN	7.03	4377	Finans Yat	IS Yat

### - NET LOT Okuma Fonksiyonları – Sistem.NetLot(...)

iDeal Sistem kütüphanesinde Net Lot olarak ekranlarda izlediğimiz verilere erişim için tanımlanmış 2 adet fonksiyon bulunmaktadır. Fonksiyonların kullanım şekilleri ve sunduğu bilgilerin neler olduğu aşağıdadki gibidir; (Not: en az düzey1 plus Borsa lisansı gerekir)

• Sistem.NetLot(Sembol):

Bu fonksiyon, herhangi bir hisse veya vadeli kontratın, bulunduğumuz gün için, bu komutu çalıştırdığımız an itibariyle LOT bazlı NET hacim bilgisini (yani alıştan olan işlemlerin lotu ile satıştan olan işlemlerin lotu arasındaki farkını) döner.

### Örnek: Ekim 2020 Vadeli Endeks30 vadeli kontratının net lotu?



var Sembol = "VIP'F\_XU0301020"; var NetLot = Sistem.NetLot(Sembol); Sistem.Mesaj(NetLot.ToString("0,000"));

× ← F_X	U030102	0 D G K	(К-					-	×
1262.00	G% 2.64	NetLot	41,569 👝						
Tavan	Yks	s ÖncK	Frk	%	Cr L	200 20			
1352.25 Tahan	1264. Del	25 1229.5	00 2.0	о4 5/ сн А	2.50 ckD	390,791	L 12	45.4700 K. Gün	
1106.75	1220	.00 1260.2	25 1229	.su A	6372	8806	n	36	
A.Saat	A.F	A.Lot	Alıs	Satis	S.L.	nt	S.F	S.Saal	ł
17.10.20			1050.00	1050.05	OIL.			47-40-7	
17:18:39	y 7	36 02	1262.00	1262.25	14	- 1	ゴ 13	17:18:3	4
17:18:37	22	184	1261.50	1262.30	85		10	17:18:3	5
17:18:33	18	177	1261.25	1263.00	12	2	19	17:18:3	17
17:18:33	27	167	1261.00	1263.25	31	2	21	17:18:3	19
17:18:33	17	204	1260.75	1263.50	203	3	26	17:18:3	9
17:18:33	23	98	1260.50	1263.75	10		16	17:18:3	1
17:10:33	20 43	145	1200.23	1264.00	15	4 5	24	17.10:3	19
17:18:35	20	114	1259.75	1264.50	28	8	34	17:18:3	9
11120133	206	1,590	1260.70	1263.65	1,57	6	186	1112013	
		29,225	1225.66	1279.59	7,19	)5			
	Pν	s	AL			S	АТ		
1	Ру	s	AL			S	AT		
No	Py Saat	s Fiyat	AL Lot	Alış	Satış	S İşlem	AT	Alan	+
No 67694	Py Saat 17:18:39	Fiyat	AL Lot	Alış :	Satış 262.25	S İşlem 662406	AT	Alan Normal	
No 67694 1 67693 1	Py Saat 17:18:39 17:18:31	Fiyat 1262.00 1262.25	AL Lot	Alış 262.00 1 262.00 1	Satış 262.25 262.25	S İşlem 662406 662140	AT	Alan Normal Normal	i
No 67694 67693 67692	Py Saat 17:18:39 17:18:31 17:18:31	s Fiyat 1262.00 1262.25 1262.25	AL Lot 1 1 17 1	Alış 262.00 1 262.00 1 262.00 1	Satış 262.25 262.25 262.25	S İşlem 662406 662140 662137	AT	Alan Normal Normal Normal Normal	ţ
No 67694 67693 67692 1 67691 67691	Py Saat 7:18:39 7:18:31 7:18:31 7:18:31 7:18:31	s Fiyat 1262.00 1262.25 1262.25 1262.25	AL Lot 1 1 17 1 5 1	Alış 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1	Satış 262.25 262.25 1262.25 1262.25	S İşlem 662406 662140 662137 662131	AT	Alan Normal Normal Normal Normal	Ì
No 67694 67693 67692 67691 57690 57690	Py Saat 7:18:39 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31	s Fiyat 1262.00 1262.25 1262.25 1262.25 1262.25 1262.25	AL Lot 1 1 17 1 5 1 1 1	Alış 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.25	S İşlem 662406 662140 662137 662138 662128 662128	AT	Alan Normal Normal Normal Normal Normal	Ĵ
No 67694 67693 67692 67691 67690 67689 67689 67689	Py Saat 17:18:39 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31	Fiyat 1262.00 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25	AL	Alış 262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25	S İşlem 662406 662140 662137 662137 662128 662125 662122	AT	Alan Normal Normal Normal Normal Normal Normal	ţ
No 67694 1 67693 1 67691 1 67691 1 67690 1 67690 1 67689 1 67688 1 67688 1 67688 1	Py Saat 17:18:39 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:23	s Fiyat 1262.00 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25	AL	Alış 1 262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.20	S İşlem 662406 662140 662137 662131 662128 662122 662122 661842	AT	Alan Normal Normal Normal Normal Normal Normal Normal	ţ
No 67694 67693 67691 67691 67691 67690 67689 67689 67688 67688 67687 67686 67686	Py Saat 7:18:39 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:23 17:18:23 17:18:22	Fiyat 1262.00 1262.25 1262.00 1262.00 1262.00	AL Lot 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Alış 262.00 1 1262.00	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.20 262.00	S İşlem 662406 662140 662137 662131 662128 662122 662122 661842 661779	AT	Alan Normal Normal Normal Normal Normal Normal Normal Normal	ţ
No 67694 67693 67691 67691 67690 67689 67688 67688 67687 67687 67687 67685	Py Saat 7:18:39 7:18:31 7:18:31 7:18:31 7:18:31 7:18:31 7:18:31 7:18:23 7:18:22 17:18:22	Fiyat 1262.00 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25 1262.25 1262.00 1262.00 1262.00	AL Lot 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Alış 1 262.00 1 1261.75 1 1261	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.00 262.00 262.00	\$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$	AT	Alan Normal Normal Normal Normal Normal Normal Normal Normal Normal	ţ
No 67694 67693 67692 67690 67690 67689 67688 67688 67687 67685 67685 67685	Py Saat 7:18:39 7:18:31 7:18:31 7:18:31 7:18:31 7:18:31 7:18:31 7:18:23 17:18:22 17:18:22 17:18:22	<ul> <li>Fiyat</li> <li>1262.00</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <l< td=""><td>AL Lot 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td><td>Alış 262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1261.75 1 1261.75 1 1261.75 1</td><td>Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.00 262.00 262.00 262.00</td><td>S İşlem 662406 662140 662137 662137 662138 662122 661842 661779 661776 661776 661773</td><td>AT</td><td>Alan Normal Normal Normal Normal Normal Normal Normal Normal Normal</td><td>Ì</td></l<></ul>	AL Lot 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Alış 262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1261.75 1 1261.75 1 1261.75 1	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.25 262.00 262.00 262.00 262.00	S İşlem 662406 662140 662137 662137 662138 662122 661842 661779 661776 661776 661773	AT	Alan Normal Normal Normal Normal Normal Normal Normal Normal Normal	Ì
No 67694 67693 67692 67690 67690 67689 67689 67687 67685 67685 67685 67685 67685 67685 67683 67685	Py Saat 7:18:39 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:23 17:18:22 17:18:22 17:18:22 17:18:22	<ul> <li>Fiyat</li> <li>1262.00</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <l< td=""><td>AL Lot 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td><td>Alış 262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1261.75 1 1261.75 1 1261.75 1 1261.75 1</td><td>Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.00 262.00 262.00 262.00 262.00</td><td>S İşlem 662406 662140 662137 662137 662138 662122 662122 661842 661779 661776 661776 661773 661770</td><td>AT</td><td>Alan Normal Normal Normal Normal Normal Normal Normal Normal Normal Normal</td><td>Ì</td></l<></ul>	AL Lot 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Alış 262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1262.00 1 1261.75 1 1261.75 1 1261.75 1 1261.75 1	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.00 262.00 262.00 262.00 262.00	S İşlem 662406 662140 662137 662137 662138 662122 662122 661842 661779 661776 661776 661773 661770	AT	Alan Normal Normal Normal Normal Normal Normal Normal Normal Normal Normal	Ì
No 67694 67693 67692 67690 67690 67689 67689 67687 67687 67685 67685 67685 67685 67684 67683 67683 67683	Py Saat 7:18:39 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:22 17:18:22 17:18:22 17:18:22 17:18:22 17:18:22 17:18:28	<ul> <li>Fiyat</li> <li>1262.00</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> </ul>	AL Lot 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Alış 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1 261.75	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.00 262.00 262.00 262.00 262.00 262.00 262.00	S İşlem 662406 662140 662137 662131 662138 662125 662122 661842 661779 661776 661779 661770 661697	AT	Alan Normal Normal Normal Normal Normal Normal Normal Normal Normal Normal	ţ
No 67694 67693 67692 67690 67690 67689 67689 67687 67687 67685 67685 67685 67685 67684 67683 67684 67683 67684 67683 67684 67683 67683 67683 67684 67683 67680 67683 67680	Py Saat 7:18:39 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:31 17:18:22 17:18:22 17:18:22 17:18:22 17:18:22 17:18:22 17:18:22 17:18:18	<ul> <li>Fiyat</li> <li>1262.00</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.25</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> <li>1262.00</li> </ul>	AL Lot 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Alış 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1 262.00 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.75 1 261.05 1 261.05 1 260.0 1 1 1 260.0 1 1 1 260.0 1 1 1 260.0 1 1 1 1 260.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Satış 262.25 262.25 262.25 262.25 262.25 262.25 262.00 262.00 262.00 262.00 262.00 262.00 262.00 262.00	S İşlem 662406 662140 662137 662131 662128 662122 661842 661779 661776 661779 661776 661770 661687 661687 661687	AT	Alan Normal Normal Normal Normal Normal Normal Normal Normal Normal Normal Normal Normal	Ì

### • Sistem.NetLotOranOran(Sembol):

Bu fonksiyon, herhangi bir hisse veya vadeli kontratın, bulunduğumuz gün, bu komutu çalıştırdığımız an itibariyle LOT bazlı NET hacim bilgisinin (yani alıştan olan işlemlerin lotu ile satıştan olan işlemlerin lotu arasındaki farkının) o hissede o gün o ana kadarki toplam lot miktarına olan oranını döner.

### - Non Linear Ehler's Filter - Sistem.NonlinearEhlersFilter(15)

Non Lineer Ehler's Filter olarak bilinen indikatörü çağırır. 1 Adet parametre alır (varsayılan 15) . Aşağıdaki gibi 2 yazım şekli vardır;

```
Sistem.NonlinearEhlersFilter(15);
Sistem.NonlinearEhlersFilter (Veriler,15);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var EF = Sistem. NonlinearEhlersFilter(15);
Sistem.Cizgiler[0].Deger = EF;
```



### - OBV – OnBalance Volume - Sistem.OnBalanaceVolume()

OBV veya On Balance Volume olarak bilinen indikatörü çağırır. İndikatör parametre almaz. Aşağıdaki gibi 2 yazım şekli vardır;

<u>NOT:</u> Bu indikatörün anlık olarak hesaplanması için Anlık olarak hacim bilgisi görme yetkisi gerekir. Bu da Borsa İstanbul Piyasaları için Düzey1 Plus (1 kademe derinlik) olarak bilinen yetkidir.

```
Sistem.OnBalanceVolume();
Sistem.OnBalanceVolume(Veriler);
```

Örnek: İndikatörü ve ortalamasını çağırıp çizdirmek

```
var OBV = Sistem.OnBalanceVolume();
var ORT = Sistem.MA(OBV, "Exp", 10);
Sistem.Cizgiler[0].Deger = OBV;
Sistem.Cizgiler[1].Deger = ORT;
```

<u>Önceki Kapanış - Sistem.OncekiKapanis...(Sembol)</u>

Bir sembolün önceki zaaman dilimlerine göre istenen kapanış fiyatını okumak için kullanılır. Fonksiyonun içine Önceki Kapanış fiyatı okutulmak istenen sembol yazılır.

<u>Not:</u> iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir. Örneğin USDTYR için FX'USDTRY şeklinde yazılır. Bir sembolün PİYASA kodunun ne olduğu, o sembolü sayfanıza yazarken @ işareti yanında gösterilir.)

Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.OncekiKapanisAltiAy (Sembol);
Sistem.OncekiKapanisBirAy(Sembol);
Sistem.OncekiKapanisBirHafta(Sembol);
Sistem.OncekiKapanisBirYil(Sembol);
Sistem.OncekiKapanisBuAy(Sembol);
Sistem.OncekiKapanisBuHafta(Sembol);
Sistem.OncekiKapanisBuYil(Sembol);
Sistem.OncekiKapanisGun(Sembol);
Sistem.OncekiKapanisSeans(Sembol);
Sistem.OncekiKapanisUcAy(Sembol);
```

- Optimizasyon - Sistem.Optimizasyon()

Bir algoritma geliştiren yatırımcı için, geliştirdiği algortima ile ilgi çok önemli 2 unsur vardır. Bunlardan birincisi BacTest (geçmiş dönemde stratejisi ne kazandırmış veya kaybettirmiş bunu görmek), ikincisi ise Opitmizasyondur. Optimizasyon, stratejide kullanılan çeşitli parametrelerin (stop veya kar al seviyeleri/oranları, kullanılan indikatörlerin periyotları, alım veya satıma karar veren ve değişken olan hemen her kriter) kullanılan değerlerinin haricinde

değerler kullanılsa idi sonuçlar nasıl olurdu, hangi değerler kullanılsa idi getiri daha yüksek olursu, işlem sayısı veya profit factor daha iyi olurdu gibi soruların cevapları Optimizasyon ile aranır.

Optimizasyon işlemi, Sistem menüsü içersinde yer alan Optimizasyon isimli pencere açılıp, bu amaçla yazılmış formülü seçerek yapılır. Yani al ve sat siyalleri üreten bir kod yazılmışsa, bu kod doğrudan optimize edilmez, önce optimize edilebilecek şekilde yine kodlama gerekir. Bir stratejinin kurgusu kabaca şu şekildedir;

- Veri ve Liste tanımlamaları yapılan satırlar (grafikten kapanış vs bilgileri okumak, gerekiyorsa/varsa boş listeleri tanımlamak)
- Kullanılacak indikatörleri tanımlamak
- Hesaplama, kıyaslama ve bazı başka sonuçları listeleri elde etme satırları
- Stratejiyi kurgulama (yön değerlerini koşullara göre atama)

Bu genel yapıda kurgulanmış bir sistemin optimizasyon kodunu yazmak için söz konusu koda 3 ana yapı entegre edilir ve sonucundan yukarıdaki strateji kurgusunun akışı aşağıdaki hale gelir;

- Veri ve Liste tanımlamaları yapılan satırlar
- Kullanılan indikatörlerde yer alan ve optimize edilerek en uygun değerleri aranacak paramatreye sahip olan her bir indikagtör için, ilgili paremetre(ler)in başlayıp biteceği değer aralıkları ve adımlarının belirtildiği for döngüleri açıp, o dönünün altında ilgili indikatörü tanımlamak ve bu işi opt edilkecek her bir indikatör veya parametre için tekrar etmek
- Strateji bölgesine gelip koşullara göre yön atamalarını yapmadan önce, her seferinde bir önceki paramatre ile doldurulmuş YÖN listesini temizlemek
- Stratejiyi çalıştırıp yön listesini doldurmak
- Sistem.Optimizasyonu komutunu programa bildirmek
- Üst kısımda açılan her bir for döngüsünü kapatmak.

ÖRNEK: 2 hareketli ortalamanın birbirini kesmesi ve RSI indikatörünün kendi hareketli ortalamasının üstünde/altında olmasıyla AL/SAT sinyalleri üreten bir strateji için ÖNCE SİSTEM sonra OPTİMİZASYON kurgusunu kodlayalım;

### **SISTEM KURGUSU:**

```
// kapanış fiyatlarını oku
var C = Sistem.GrafikFiyatSec("Kapanis");
var SonYon="";
//indikatörleri tanımla
var MA1 = Sistem.MA(C , "Exp", 10);
var MA2 = Sistem.MA(C , "Exp", 15);
var RSI = Sistem.RSI(C , 9);
var RSIAVR = Sistem.MA(RSI , "Exp", 4);
//Strateji bölgesi
for (int i = 1; i < C.Count; i++)
{
    if (RSI[i] > RSIAVR[i] && MA1[i] > MA2[i] && SonYon != "A") // alış
    {
        Sistem.Yon[i] = "A";
        SonYon="A";
    }
}
```

```
}
    else if (RSI[i] < RSIAVR[i] && MA1[i] < MA2[i] && SonYon != "S") // satış
    {
        Sistem.Yon[i] = "S";
        SonYon="S";
    }
}</pre>
```

### **OPTIMIZASYON KURGUSU:**

```
// kapanış fiyatlarını oku
var C = Sistem.GrafikFiyatSec("Kapanis");
var SonYon="";
for (int P1 = 10; P1 < 101; P1++) // MA1 için periyodu 10 ile 101 arasında birer birer adımlarla
{
   var MA1 = Sistem.MA(C , "Exp", P1); //yukarıdaki afdımların her biri için MA hesaplansın
   for (int P2 = 15; P2 < 250; P2++)
   {
      var MA2 = Sistem.MA(C , "Exp", P2);
      for (int P3 = 9; P3 < 59; P3++)
      {
         var RSI = Sistem.RSI(C , P3);
         for (int P4 = 4; P4 < 44; P4++)
         {
            var RSIAVR = Sistem.MA(RSI , "Exp", P4);
               //ÖNCEKİ DÖNGÜLERDEN KALAN YÖN LİSTESİNİ TEMİZLE VE DEVAM ET
               for (int i = 1; i < C.Count; i++)</pre>
                  Sistem.Yon[i] = "";
            // strateji bölgesi
            for (int i = 1; i < C.Count; i++)</pre>
            {
               if (RSI[i] > RSIAVR[i] && MA1[i] > MA2[i] && SonYon != "A") // alış
               {
                  Sistem.Yon[i] = "A";
                  SonYon="A";
               }
               else if (RSI[i] < RSIAVR[i] && MA1[i] < MA2[i] && SonYon != "S") // satış
               {
                  Sistem.Yon[i] = "S";
                  SonYon="S";
               }
             Sistem.Optimizasyon("MA1,MA2,RSI,AVR", P1 , P2 , P3 , P4);
//açılan for dmngüleri kapat (kapa süslü poarantezler)
          }
       }
   }
}
```

### - Ortalama Fiyat - Sistem.Ortalama...(Sembol)

Bir sembolün herhangi bir zaman dilimine ait ortalama fiyatını okumak için kullanılır. Fonksiyonun içine Ortalama Fiyatı okutulmak istenen sembol yazılır.

<u>Not:</u> iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir. Örneğin USDTYR için FX'USDTRY şeklinde yazılır. Bir sembolün PİYASA kodunun ne olduğu, o sembolü sayfanıza yazarken @ işareti yanında gösterilir.)

Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.OrtalamaAltiAy(Sembol);
Sistem.OrtalamaBirAy(Sembol);
Sistem.OrtalamaBirHafta(Sembol);
Sistem.OrtalamaBirYil(Sembol);
Sistem.OrtalamaBuAy(Sembol);
Sistem.OrtalamaBuHafta(Sembol);
Sistem.OrtalamaBuYil(Sembol);
Sistem.OrtalamaGun(Sembol);
Sistem.OrtalamaSeans(Sembol);
Sistem.OrtalamaUcAy(Sembol);
```

### - OtoTrend Düşen - Sistem.OtoTrendDusen(TopmaBar, SonXbar)

Bir grafik üzerinde, kullanıcı tarafından girilmiş kadar bar dilimi içerisindeki DÜŞEN TRENDİ hesaplar. 2 parametre girilerek kullanılır. ToplamBar paramatresi grafikte SON KAÇ BAR İÇİNDE düşen bir trend aranacağını, SonXbar paramatresi ise, grafikte en sondaki kaç barın bu trend hesabında devre dışı bırakılacağını belirtir. EN SON BAR DAHİL trend aranmaz, çünkü zaten son bar yükselen veya düşen trendin bir noktası olur ise, söz konusu trend için kıranlar gibi bir tarama yapılamaz.

Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.OtoTrendDusen(ToplamBar, SonBar);
Sistem.OtoTrendDusen(Veriler, ToplamBar, SonBar);
```

İlk kullanım şekli, üzerine uygulandığı grafiğin sembol, periyot bilgilerini kendi algılar. Bu komutları grafiğe uygulamadan (örneğin bir çok hissede düşen trend kırılımı arayan bir robotta) kullanmak istersek o zaman ikinci yazım şekli kullanılır. Veriler değişkenine hisse ve zaman periyodu belirtilerek okutulmuş bir veri listesi input olarak verilir.

ÖRNEK: Son 1000 ve son 300 bar içindeki düşen trenleri bul ve çiz

```
var UZUN = 1000;
var KISA = 300;
var SON = 5;
var DusenUzun = Sistem.OtoTrendDusen(UZUN, SON);
var DusenKisa = Sistem.OtoTrendDusen(KISA, SON);
Sistem.Cizgiler[0].Deger = DusenUzun;
Sistem.Cizgiler[1].Deger = DusenKisa;
```

Yukarıdaki formülü kaydedip bir grafiğe uyguladığınızda, istediğiniz sembolün istediğiniz zaman periyotlu grafiğine geçererek, eğer varsa belirtilen zaman dilimleri için düşen trenler otomatik hesaplanıp çizilir.



### OtoTrend Yükselen - Sistem.OtoTrendYukselen(TopmaBar, SonXbar)

Bir grafik üzerinde, kullanıcı tarafından girilmiş kadar bar dilimi içerisindeki YÜKSELEN TRENDİ hesaplar. 2 parametre girilerek kullanılır. ToplamBar paramatresi grafikte SON KAÇ BAR İÇİNDE yükselen bir trend aranacağını, SonXbar paramatresi ise, grafikte en sondaki kaç barın bu trend hesabında devre dışı bırakılacağını belirtir. Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.OtoTrendDusen(ToplamBar, SonBar);
Sistem.OtoTrendDusen(Veriler, ToplamBar, SonBar);
```

İlk kullanım şekli, üzerine uygulandığı grafiğin sembol, periyot bilgilerini kendi algılar. Bu komutları grafiğe uygulamadan (örneğin bir çok hissede düşen trend kırılımı arayan bir robotta) kullanmak istersek o zaman ikinci yazım şekli kullanılır. Veriler değişkenine hisse ve zaman periyodu belirtilerek okutulmuş bir veri listesi input olarak verilir.

### ÖRNEK1: Son 1000 ve son 300 bar içindeki yükselen trenleri bul ve çiz

```
var UZUN = 1000;
var KISA = 300;
var SON = 5;
var YukselenUzun = Sistem.OtoTrendYukselen(UZUN, SON);
var YukselenKisa = Sistem.OtoTrendYukselen(KISA, SON);
Sistem.Cizgiler[0].Deger = YukselenUzun;
Sistem.Cizgiler[1].Deger = YukselenKisa;
```

Yukarıdaki formülü kaydedip bir grafiğe uyguladığınızda, istediğiniz sembolün istediğiniz zaman periyotlu grafiğine geçererek, eğer varsa belirtilen zaman dilimleri için yükselen trendler otomatik hesaplanıp çizilir.



### ÖRNEK2: Yükselen ve Düşen Trendi Kıran hisseleri SORGU modülünde taratıp bulmak:

```
Sistem.SorguBaslik[0] = "Fiyat";
Sistem.SorguBaslik[1] = "Onc.Kap";
var Trendperiyodu = 800; //son 800 bar içindeki trend
var SonXbar= 50; //son 50 bar1 dikkate alma
var Sembol = Sistem.Sembol;
var V = Sistem.GrafikVerileriniOku(Sembol, Sistem.Periyot);
var C = Sistem.GrafikFiyatOku(V,"Kapanis");
var Yukselen = Sistem.OtoTrendYukselen(V, Trendperiyodu , SonXbar);
var Dusen = Sistem.OtoTrendDusen(V, Trendperiyodu , SonXbar);
if (V[V.Count - 1].Close < V[V.Count - Trendperiyodu].Close && V[V.Count - 1].High >
Dusen[Dusen.Count - 1] && Dusen[Dusen.Count - 1] != 0)
{
   Sistem.SorguDeger[0] = C[C.Count - 1];
   Sistem.SorguDeger[1] = C[C.Count - 2];
   Sistem.SorguAciklama = "Düşen Trend Kırıldı";
   Sistem.SorguEkle();
}
else if (V[V.Count - 1].Close > V[V.Count - Trendperiyodu].Close && V[V.Count - 1].Low <
Yukselen[Yukselen.Count - 1] && Yukselen[Yukselen.Count - 1] != 0)
{
   Sistem.SorguDeger[0] = C[C.Count - 1];
   Sistem.SorguDeger[1] = C[C.Count - 2];
   Sistem.SorguAciklama = "Yükselen Trend Kırıldı";
   Sistem.SorguEkle();
}
```

### - Özel Emirleri Oku - Sistem.OzelEmirleriGetir()

Borsa İstanbul Pay Piyasasında özel emir olarak bazen hisse senetleri el değiştirir. Bu emirler TÜM İŞLEMLER isimli pencereden akarken TİP alanında ÖZEL yazar ve buradaki işlemler bazen o anki piyasa fiyatının çok dışında fiyatlarla hisselerin el değiştirdiği birer işlemdir. Bu işlemlerin fiyatları grafiklerde çizilmez. O gün özel emir var mı, varsa hangi hisselerde hangi kurumlar hangi fiyattan hisse alış verişi yaptı sorunuzun cevabı için BIST menüsündeki TÜM

İŞLEMLER penceresini açmak, FİLTRE satırına basıp ÖZEL kutusunu işaretleyip uygula demek yeterlidir.

im > Filtre	
🔽 Özel	Kurum Seçiniz 👻
TE	📄 Alışlar
NSE	📃 Satışlar
Uygula	Tüm Kapat

Örneğin bu klavuzun yazıldığı gün aşağıdaki özel emirler gerçekleşmiştir.

🗶 🔶 Bist	Tüm Semb	oller Bug	ün Ack	Hacim ≥ 0	Filtre			-
Hacim 32,530,18	Para 8,288 17,3	a + 392,549,888	Para- 15,137,6	Net 38,400 2,25	4,911,488	Net% 6.93		
No	İşlem	Saat	Senet	Fiyat	Lot	π	Alan	Satan
4 3 2 1	A3995810 A3995820 A3995820 A3995820 A3995820	17:42:45 11:08:36 11:04:59 11:00:25	GARAN VESTL VESTL VESTL	9.72 19.82 19.82 19.82 19.82	1500000 300000 1000000 900000	14580000 5946000 19820000 17838000	Is Yat HSBC YFAS Gedik	YapiKredi Deniz Deniz Deniz

Bu özel emir bilgilerini SİSTEM üzerinden erişmek için ÖZEL EMİRLERİ OKU fonksiyonu vardır. Fonksiyonun kullanımını içeren örnek bir kod aşağıda paylaşışmıştır;

Bu kod kaydedilip bir kez çalıştırıldığında (FORMÜL TEST butonuna basarak) ekrana o ana kadar gerçekleşmiş özel emirlere ait detayları gösteren bir mesaj penceresi açılır.;

```
if (DateTime.Now.Hour < 19)
{
    var OE = Sistem.OzelEmirleriGetir();
    if (OE != null)
    {
       var Anahtar = Sistem.Name + "," + DateTime.Now.ToString("yyyyMMdd");
       var Adet = Sistem.SayiTablosunuOku(Anahtar);
        if (OE.Count != Adet)
        {
          Adet = OE.Count;
          Sistem.SayiTablosunuGuncelle(Anahtar, Adet);
          var strlist = new List<string>();
          foreach (var item in OE)
          {
var str = item.Symbol + " , " + "Saat = " + item.Hour.ToString("") + ":" +
item.Minute.ToString("") + ":" + item.Second.ToString("") + " , " + "Fiyat = " +
item.Price.ToString("0.00") + " , " + "Lot = " + item.Size.ToString() + " Alan = " + item.BuyerCode
+ " Satan= " + item.BuyerCode;
              strlist.Add(str);
          var msg = string.Join("\r\n", strlist);
          Sistem.Mesaj(msg);
       }
   }
}
```



OzelEmirTakip	
VESTL , Saat = 11:0:25 , Fiyat = 19.82	, Lot = 900000 Alan = GDK Satan= GDK
VESTL , Saat = 11:4:59 , Fiyat = 19.82	, Lot = 1000000 Alan = YAT Satan= YAT
VESTL , Saat = 11:8:36 , Fiyat = 19.82	, Lot = 300000 Alan = HSY Satan= HSY
GARAN , Saat = 17:42:45 , Fiyat = 9.72	, Lot = 1500000 Alan = IYM Satan= IYM

- PSAR / PARABOLIC - Sistem.Parabolic(0.02, 0.2)

PSAR, Parabolis SAR veya Parabolic olarak bilinen indikatörü çağırır. 2 Adet parametre alır (varsayılan olarak 0.02 ve 0.2 kullanılır). Aşağıdaki gibi 3 yazım şekli vardır;

```
Sistem. Parabolic(0.02, 0.2);
Sistem. Parabolic(Liste,0.02, 0.2);
Sistem. Momentum(Veriler,0.02, 0.2);
```

Örnek: Fiyat Parabolic indikatörünü yukarı keserse al, aşaüğı keserse SAT sistem formülü.

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var Parabolic = Sistem.Parabolic(0.02, 0.2);
var SonYon = "";
for (int i = 1; i < Sistem.BarSayisi; i++)</pre>
{
   if (C[i-1] < Parabolic[i-1] && C[i] > Parabolic[i] && SonYon != "A")
   {
      SonYon = "A";
      Sistem.Yon[i] = SonYon;
   }
   if (C[i-1] > Parabolic[i-1] && C[i] < Parabolic[i] && SonYon != "S")</pre>
   {
      SonYon = "S";
      Sistem.Yon[i] = SonYon;
   }
}
Sistem.Cizgiler[0].Deger = Parabolic;;
```





- Parametreler - Sistem.Parametreler

Parametreler fonksiyonu, bazen kodu sadeleştirmek, bazen kod yazımını kolaylaştırıp anlaşılır kılmak, bazen robot olarak çalıştırılan kodların bazı sonuçlar üzerinden gözlem yapmak, bazen şifrelenmiş olarak sunulmuş/dağıtılmış formüllere, dışardan müdahaleye izin ve imkân vermek için kullanılabilir.

IDEAL sistem panelininin (kod yazılan pencere) sağ üst bölgesinde 20 satırlık bir PARAMETRE bölgesi vardır.

Bu fonksiyonun işlevi, bu satırlara yazılan verileri koda aktarmak veya koddan elde edilecek bazı bilgileri bu alanlara yazmaktır.

Sistem.Parametreler[x] şeklinde kullanılır. X değeri parametre bölgesindeki satırların numarasıdır. İLK SATIRIN NUMARASI SIFIRDIR.

Parametreler kısmında bir satıra yazılmış bir veri bu fonksiyon ile okutulup, kod içinde kullanılabilir. (aşağıdaki örnek)

van v - Cistom Ranamatnalan[0]:							
Sistem.Mesaj(x);	No	Parametre					
	0	Bu Bir Denemedi					
	1						
🖳 iDeal Mesai 09.10.2015 11:55:47	2						
· · · · · · · · · · · · · · · · · · ·	3						
Bu Bir Denemedi							

Yukardaki işlemin tersi (kodun belli bir anında elimizde olan bir veri, parametreler kısmındaki bir satıra yazılabilir.) yapılabilir. (aşağıdaki örnek: Aktif VIOP kontratının Son Fiyat değeri okutulup Parametreler tablosunun 2 numaralı (3. Sıra) satırına yazdırılmıştır.) (Mesaj ve Debug fonksiyonlarında anlatıldığı gibi, tipi ne olursa olsun, ekranda görsel olarak gösterilmek istene bir veri, sonuna **.ToString()** ibaresi eklenerek (metin/text) şekline çevrilir.)

Parametre fonksiyonu genelde iki önemli amaca hizmet eder.

- 1- Yazılan formüllerde kullanılan ve zaman zaman değiştirilmesi gereken (indikatör parametreleri, kırılma noktaları için fiyat seviyeleri, sözleşme kodları gibi) bilgileri bu alandan okutarak, değişiklik gerektiğinde kolayca yapmak.
- 2- Çeşitli amaçlarda, yazdıkları formülleri bir başkasına ŞİFRELEYEREK veren kişilerin, kodu verdiği kişiye bazı bilgileri değiştirebilme imkânı vermek istemesi. (Kodlar şifrelendiğinde bu alan açık kalır)

ÖRNEK: GARAN hisse senedinde, manuel olarak Pozisyon açan bir kullanıcının, kendi belirlediği ve zaman zaman değiştirdiği yükseliş ve düşüş stop seviyelerini girdiği ve o seviyelere gelindiğinde otomatik olarak pozisyon kapatmasını sağlayan ROBOT örneği.

İşlem adedi, pozisyona girilmiş fiyat ve yön ile stop seviyeleri parametre alanına girilir. Robot pozisyonu kapattığı zaman (stop seviyeleri görüldüğünde) Parametreler bölgesindeki YÖN bilgisini kendisi değiştirir.

```
string Sembol
                        = "IMKBH'GARAN";
float IslemFiyati
                        = Convert.ToSingle(Sistem.Parametreler[1]); //Pozisyona girilmis
olan fiyat
float KarMarj
                        = Convert.ToSingle(Sistem.Parametreler[3])/100;
                                                                            //Stop
                                                                                      için
beklenecek artış puanı
                        = Convert.ToSingle(Sistem.Parametreler[5])/100;
                                                                            //Stop
                                                                                      için
float ZararMari
beklenecek düsüs puanı
float Miktar
                        = Convert.ToSingle(Sistem.Parametreler[7]); //Alış/Satış emri için
Lot Miktarı
                        = Sistem.Parametreler[9]; // Girilen pozisyonun yönü belirlenir
string Pozisyon_Yonu
Örnek: Long=L Short=S Flat= F
var Islem = "";
var SonFiyat = Sistem.SonFiyat(Sembol);
var Pozisyon = Sistem.PozisyonKontrolOku(Sembol);
if (SonFiyat > 0 && (Sistem.SaatAraligi("09:35", "12:30") || Sistem.SaatAraligi("13:00",
"17:40")))
{
   if (Pozisyon_Yonu == "L" && (SonFiyat >= IslemFiyati + KarMarj) && Pozisyon > 0)
      Islem = "SATIS";
   else if (Pozisyon Yonu == "S" && (SonFiyat <= IslemFiyati - ZararMarj) && Pozisyon < 0)
      Islem = "ALIS";
}
if (Islem != "" && Pozisyon_Yonu != "F")
.
Sistem.Parametreler[9] = "F"; //Pozisyon Kapatıldığı için Parametre Değerine Flat ifadesi
atanır.
if (Islem == "ALIS")
      Sistem.PozisyonKontrolGuncelle(Sembol, 0);
   else if (Islem == "SATIS")
      Sistem.PozisyonKontrolGuncelle(Sembol, 0);
   Sistem.EmirSembol = Sembol;
   Sistem.EmirIslem = Islem;
   Sistem.EmirMiktari = Miktar;
   Sistem.EmirSuresi = "KIE"
   Sistem.EmirTipi = "Piyasa";
   Sistem.EmirGonder();
}
```

### - Periyot - Sistem.Periyot()

Üzerinde işlem yapılan veya yapılacak olan grafiğin periyodunu okumak veya belirtmek için bu fonksiyon kullanılır.

Yazılan bir kodun/sistemin periyodu sabit değilse ve diğer periyotlara geçince de çalışılması isteniyorsa, bu fonksiyonu kullanmaya gerek yoktur.

Ama grafik üzerinden değiştirilen bir periyoda göre bir başka işlem (mesela aynı periyot için bir başka sembolün herhangi bir verisi de okutuluyorsa) o zaman periyodun ne olduğunu anlamak ve diğer işlemlerimizde kullanmak için bu fonksiyon kullanılır.

Grafik üzerinden çalışmayan bir kod yazılmış ise (Algo veya Robot) ve bir grafik verisi kodlamada kullanılıyorsa, periyot mutlaka kodda belirtilmelidir.

IDEAL Teknik Analiz modülünde, bir kodun grafiği aşağıda gösterilen periyotlar için

# ΠΕΔL

görüntülenebilir. Sistem modülünde kod yazarken bu periyotların tamamı kullanılabilir. Sistemde kullanılacak periyot değerleri, aşağıdaki fotoda SAĞ TARAFTA yer alan kısa kullanım şeklinde olmalıdır.

Giin		T Darabolic	
	Birim	В	
	F	F	
	R	R	
	5 sn	55	
	10 sr	10S	
	15 sr	<b>1</b> 5S	
	1 dk	1	
	2 dk	2	
	3 dk	3	
	4 dk	4	
4	5 dk	5	
	8 dk	8	
	10 di	10	
	15 di	15	
	20 di	20	
	30 di	30	
	60 dl	60	
4	120 c	< 120 I	
2	240 c	< 240	
	Sean	s	
<b>~</b>	Gün	G	
N	Haft	н	
E V	Ay	А	
ĸ	3 Ay	U	
K	Yil		

Periyot kontrolünü grafik üzerine atılan kod içinde yaparak, örneğin çalışılması istenen grafik periyodundan başka bir periyoda geçince kodun çalışmasına engel olabilir ve ekrana bilgi/uyarı mesajı çıkarabilirsiniz.

ÖRNEK:

```
var periyot = (Sistem.Periyot == "60" || Sistem.Periyot == "120" || Sistem.Periyot == "S"
|| Sistem.Periyot == "240" || Sistem.Periyot == "G" || Sistem.Periyot == "H" ||
Sistem.Periyot == "A" ) ? false : true;
if (periyot == true)
{
    // yazılacak kod
```



PH01 - Sistem.SembolTanimla(SembolAdı, OndalıkBasamakSayısı)

Grafikler üzerine saatlik, günlük, haftalık, aylık gibi dönemlere EN YÜKSEK / EN DÜŞÜK fiyat seviyelerini atmak için kullanılan PH01 indikatörünü kod üzerinden çağırıp kullanmak için var olan ideal sistem fonksiyonudur. Sadece HIGH çizgisini getirir. Aşağıdaki PL01 ile birlikte kullanıldığında HIGH/LOW bandı çizdirilebilir.

Aşağıda gibi 2 kullanım şekli vardır;

```
var PH = Sistem.PH01(Donem);
var PH = Sistem.PH01(Veriler , Donem);
```

Dönem ifadesi hangi zaman dilimimi HIGH seviyesini kullanmak istediğimizi seçmek için kullanılır. Yukarıda anlatılan PERİYOT fonksiyonunda gösterilen zaman dilimleri kısa harf/kod bilgileri burada dönem olarak girilen bilgidiriç. Örneğin HAFTALIK YÜKSEK seviyesi kullanılacaksa dönem için "H" yazılır.

- PL01 - Sistem.SembolTanimla(SembolAdı, OndalıkBasamakSayısı)

Grafikler üzerine saatlik, günlük, haftalık, aylık gibi dönemlere EN YÜKSEK / EN DÜŞÜK fiyat seviyelerini atmak için kullanılan PH01 indikatörünü kod üzerinden çağırıp kullanmak için var olan ideal sistem fonksiyonudur. Sadece LOW seviyesini hesaplayıp getirir.

Aşağıda gibi 2 kullanım şekli vardır;

```
var PH = Sistem.PH01(Donem);
var PH = Sistem.PH01(Veriler , Donem);
```

Dönem ifadesi hangi zaman dilimimi HIGH seviyesini kullanmak istediğimizi seçmek için kullanılır. Yukarıda anlatılan PERİYOT fonksiyonunda gösterilen zaman dilimleri kısa harf/kod bilgileri burada dönem olarak girilen bilgidiriç. Örneğin AYLIK YÜKSEK seviyesi kullanılacaksa dönem için "A" yazılır.

Örnek, haftalık ProHigh / ProLow değerlerini çağırıp çizdirmek;

```
var PH = Sistem.PH01("H");
var PL = Sistem.PL01("H");
Sistem.Cizgiler[0].Deger = PH;
Sistem.Cizgiler[1].Deger = PL;
```



### - Pivot Kanalı Sistem.Pivot(SembolAdı, OndalıkBasamakSayısı)

Hazır indikatörler listesinde yer alan alt/üst/orta ban şeklinde 3 çizgisi olan pivot isimli indikatörü çağırıp hesaplayan Deal Sistem fonksiyonudur. Aşağıdaki şekilde 6 kullanım şekli vardır;

```
Sistem.PivotUp();
Sistem.PivotUp(Veriler);
Sistem.PivotMid();
Sistem.PivotMid(Veriler);
Sistem.PivotDown();
Sistem.PivotDown(Verileri);
```

Grafik üzerine atılarak kullanılacaksa VERİLER bilgisi olan komutlara gerek yoktur. Grafik üzerinde çalıştırılmayan bir fomülde kullanılacaksa hangi sembolün hangi zaman periyotlu grafik verisi için hesaplanacağı fonksiyona söylenmelidir;

ÖRNEK: YKBNK hissesinin GÜNLÜK barları için Pivot hesapla.

```
var Veriler = Sistem.GrafikFiyatOku("IMKBH'YKBNK", "G");
var UstKanal = Sistem.PivotUp(Veriler);
var OrtaKanal = Sistem.PivotMid(Veriler);
var AltKanal = Sistem.PivotDoen(Veriler);
```

- Polarized Fractal Efficiency - Sistem.PolarizedFractalEfficiency(14,3)

iDeal indikatör kütüphanesinde yer alan Polarized Fraactal Efficiency isimli indikatörü çağırır. 2 Adet parametre alır (varsayılan 14 ve 3). Aşağıdaki gibi 2 yazım şekli vardır;



Sistem.PolarizedFractalEfficiency(14, 3); Sistem.PolarizedFractalEfficiency(Veriler, 14, 3);

#### Örnek: İndikatörü çağırıp çizdirmek

```
var PFE = Sistem.PolarizedFractalEfficiency(14, 3);
Sistem.Cizgiler[0].Deger = PFE;
```

#### - Positive Volume Index - Sistem.PositiveVolumeIndex()

iDeal indikatör kütüphanesinden yer alan Positive Volume Index isimli indikatörü çağırır. Hiç paramatre almaz. Sorunsuz ve anlık çaşlışması için HACİM anlık verisi gerektirir. Aşağıdaki gibi 2 yazım şekli vardır;

```
Sistem.PositiveVolumeIndex();
Sistem.PositiveVolumeIndex(Veriler);
```

<u>Örnek: İndikatörü çağırıp çizdirmek</u>

```
var PVI = Sistem.PositiveVolumeIndex();
Sistem.Cizgiler[0].Deger = PVI;
```

Pozisyon Kontrol Oku/Güncelle - Sistem.PozisyonKontrolOku/Guncelle()

iDeal veri terminali üzerinde robotlar çalıştırıken çoğu zaman bazı bilgileri bir dosyada veya bellekte saklamak ve gerektiğinde kullanmak gerekir. Kod yazılırken bilgi atanan (var x = 5; şeklinde tanımlanan) değişkenler bazen bu ihtiyacı karşılamaz.

Çünkü kod içerisinde tanımlanan değişkenler, kodun her seferinde dafalarca çalışması durumunda hep ilk tanımladığı değerine döner ve ilerleyen satırlarda hesaplamalar sonucu yeni değeri bulunur. Ama bu döngü her seferinde yeniden olur. Oysa istenen şey, bir kez değeri değiştirilen bir verinin, bir daha kullanıcının tanımladığı bir koşul olmadan hiç değişmemesidir. Bu nedenle iDealde Pozisyon Kontrol, Sayı Tablosu, Sözcük Tablosu gibi veri saklama işlevi olan komutlar kullanılır.

Pozisyon Kontrol genelde alım satım yapılan robotlarda kullanılır. Örneğin bir hissenin fiyatı x olursa AL kurgusuyla çalışan bir robotta, hissenin fiyatı x olduğunda ALIM yapılınca, hisse değeri düşüp sonra tekrar x olunca yeniden alım yapmaması için ilk alımda pozisyon değeri mesela 1 yapılır. Bu sayede koşul tekrar sağlansa bile robot alım yapmaz, taa ki kullanıcı pozisyon ismiyle saklanan değişkendeki değeri başka bir koşulla değiştirinceye kadar.

PozisyonKOntrolOku komutu saklanmış bilgiyi okumak için, PozisyonKontrolGuncelle komutu ise buraya bir bilgiyi yazmak için kullanılır.

Robotlar aksi kodlanmadıkça PİYASA EMRİ ile emir iletirler. Bu sayede emrin gerçekleşmesi (hisse tavan değilse, devre kesmemişse) kesindir. Emir iletilen işlem miktarı kadar pozisyona girilmiş olur ve kendi pozisyonunu bu şekilde takip eder. (aracı kurumunuzdaki hesabınızda o hisse için kaç adet pozisyonunun olduğuna değil kendi pozisyon değerine bakar).

ÖRNEK: Saat 11:01:00 olunca 1000 adet DOHOL alış emri gönder. Saat 15:01:00 olunca aldığı (pozisyonu) kadar SATIŞ emri gönder robotu;

```
var Sembol = "IMKBH'DOHOL";
var Pozisyon = Sistem.PozisyonKontrolOku(Sembol);
var Miktar = 1000;
if (Sistem.Saat.CompareTo("11:01:00") >= 0 && Pozisyon == 0)
{
   Sistem.PozisyonKontrolGuncelle(Sembol, Pozisyon + Miktar);
   Sistem.EmirSembol = Sembol;
   Sistem.EmirIslem = "Alış";
   Sistem.EmirTipi = "Piyasa";
   Sistem.EmirSuresi = "KIE"
   Sistem.EmirMiktari = Miktar;
   Sistem.EmirGonder();
if (Sistem.Saat.CompareTo("15:01:00") >= 0 && Pozisyon == Miktar)
{
   Sistem.PozisyonKontrolGuncelle(Sembol, 0);
   Sistem.EmirSembol = Sembol;
   Sistem.EmirIslem = "Satış";
   Sistem.EmirTipi = "Piyasa";
   Sistem.EmirSuresi = "KIE";
   Sistem.EmirMiktari = Pozisyon;
   Sistem.EmirGonder();
}
```

ÖRNEK2: Birden çok robot aynı hissede alış/satış yapıyor. Bütün robotların o hissedeki pozisyon değerlerini okuyabiliriz. Bu sayede mesela robotun o hissedeki (veya vadeli kontrattaki) pozisyonu x adet değilse, pozisyonu x adete tamamlayacak kadar ALIŞ veya SATIŞ emri gönderecek bir başka robot yazabiliriz.

Başka robotların bir VIOP Endeks Yakın vade kontratındaki pozisyonları aşağıdaki şekilde okunabilir;

```
var Poz1 = Sistem.PozisyonKontrolOku("Robot1" + " , " + Sistem.AktifViopKontrat);
var Poz2 = Sistem.PozisyonKontrolOku("Robot2" + " , " + Sistem.AktifViopKontrat);
var Poz3 = Sistem.PozisyonKontrolOku("Robot3" + " , " + Sistem.AktifViopKontrat);
var Poz4 = Sistem.PozisyonKontrolOku("Robot4" + " , " + Sistem.AktifViopKontrat);
var Poz5 = Sistem.PozisyonKontrolOku("Robot5" + " , " + Sistem.AktifViopKontrat);
```

- Pozisyon Kontrol komutuyla yazılan/okunan bilgiler iDeal kapanıp açılsa bile silinmez. Bu bilgiler \iDeal\Config klasörü altındaki dosyada kaydedilir ve saklanır.
- Pozisyon kontrol komutuyla istenirse işleme girildiği anın tarih saat bilgisi de kaydedilebilir.
- Pozisyon kontrol komutuyla istenirse emir üretildiği anda o hissenin herhangi bir fiyat bilgisinin ne olduğu da kaydedilebilir.
- Pozisyon kontrol komutuyla, emir iletildiği an için kullanıcı tarafından belirlenecek herhangi bir not (örneğin KAR AL YAPILDI) yazdırılabilir.

Portföy penceresinin ROBOT sekmesinde yer alan POZİSYON KONTROL başlığında, çalışan her bir robotun işlem yatığı hissede/kontratta kaç adet pozisyıonda olduğu, son pozisyoın değişim zamanı ve o zamanda fiyatın ne olduğu gibi bilgiler takip edilebilir.



iDeal Por	tföy													
			•	1	•	Login	Çıkış		Hesaplar	Ayarlar	Renk	Excel	Uyarı	Güncelle
BIST Portföy	BIST Emirler	BIST K/Z	BIST Ekstre	BIST Sepet	VIOP Portföy	VIOP Emirler	VIOP Gün İçi Özet	VIOP Nakit	R	вот				
Aksiya	Aksiyon : O Yok O Sanal O Gerçek Robot_YeniKalip_YeniPoz2													
Ay	Ayarlar Pozisyon Kontrol Emirler													
-			Ana	ncar			Pozisyo	n	Fiyat		Tarih	Y		Rezerv
Robot	Robot_YeniKalip_YeniPoz2,IMKBH'GARAN						-1		9,8700	2020	-12-20 19:02:	LO SATIŞ YAI	PILDI	
								Л						

Bu ekstra bilgileri de yazma ve okuma işlevini yerine getiren POZİSYON KONTROL komutu kullanımlarını içeren örnek bir ROBOT formülü aşağıdaki gibi yazılabilir.

```
var LotSize = 1; //işlem adedi
var SistemAdi = "MA500"; //sistemizin adı
var GrafikSembolu = "IMKBH'GARAN"; //sistemin sinyal ürettiği grafik sembolü
var GrafikPeriyodu = "1"; //grafiğin priyodu
var EmirSembol = "IMKBH'GARAN";
var MySistem = Sistem.SistemGetir(SistemAdi, GrafikSembolu , GrafikPeriyodu ); //sistemin
adı, grafik sembolü, grafiğin periyodu
var SonFiyat = Sistem.SonFiyat(EmirSembol);
var Anahtar = Sistem.Name + "," + EmirSembol;
double IslemFiyat = 0;
DateTime IslemTarih;
var Miktar = 0.0;
var Rezerv = "";
var Pozisyon = Sistem.PozisyonKontrolOku(Anahtar, out IslemFiyat, out IslemTarih);
var SonYon = Sistem.SonYonGetir(SistemAdi, GrafikSembolu , GrafikPeriyodu ); //sistemin
adı, grafik sembolü, grafiğin periyodu
if (Sistem.Saat.CompareTo("10:00:00") <= 0 || Sistem.Saat.CompareTo("20:59:59") >= 0) //
seans yok işlem yapma
{
}
else
{
   if (SonYon == "F" && Pozisyon != 0) // Flata Geç
      Miktar = -Pozisyon;
   else if (SonYon == "A" && Pozisyon != LotSize) // Al
      Miktar = LotSize - Pozisyon;
   else if (SonYon == "S" && Pozisyon != -LotSize) // Sat
      Miktar = -LotSize - Pozisyon;
    // Emir Gönder
    var Islem = ""
    if (Miktar > 0) {Islem = "ALIS"; Rezerv = "ALIS YAPILDI";}
    if (Miktar < 0) {Islem = "SATIS"; Rezerv = "SATIS YAPILDI";}</pre>
    if (Islem != "")
    {
        Sistem.PozisyonKontrolGuncelle(Anahtar, Miktar + Pozisyon, SonFiyat, Rezerv);
        Sistem.EmirSembol = EmirSembol ;
        Sistem.EmirIslem = Islem;
        Sistem.EmirSuresi = "KIE'
        Sistem.EmirTipi = "Piyasa";
        Sistem.EmirMiktari = Math.Abs(Miktar);
        Sistem.EmirGonder();
    }
}
Price Channel Up/Down - Sistem.PriceChannelDown/Up(10)
```

iDeal indikatör kütüphanesinde yer alan Price Channel (Fiyat Kanalı) isimli indikatörü çağırır. Bir adet paramatre alır ve varsayılan değeri 10 olarak kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.PriceChannelDown(10);
Sistem.PriceChannelDown(Veriler, 10);
Sistem.PriceChannelUp(10)
Sistem.PriceChannelUp(10);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var PC_DOWN = Sistem. PriceChannelDown(10);
var PC_UP = Sistem. PriceChannelUp(10);
Sistem.Cizgiler[0].Deger = PC_DOWN;
Sistem.Cizgiler[1].Deger = PC_UP;
```

- Price Oscillator Percent - Sistem.PriceOscPercent(10, 30, "Exp")

iDeal indikatör kütüphanesinde yer alan Price Oscillator Percent isimli indikatörü çağırır. Üç adet paramatre alır ve varsayılan değerleri 10/30 ve "Exp" olarak kullanılır. PERCENT kelimesi, fiyat osilatörü indikatörünü YÜZDE değişim kullanarak çağırdığımızı belirtir. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.PriceOscPercent(10, 30,"Exp");
Sistem.PriceOscPercent(Veriler, 10, 30,"Exp");
```

```
- Price Oscillator Point - Sistem.PriceOscPoint(10, 30, "Exp")
```

iDeal indikatör kütüphanesinden yer alan Price Oscillator Point isimli indikatörü çağırır. Üç adet paramatre alır ve varsayılan değerleri 10/30 ve "Exp" olarak kullanılır. POINT kelimesi, fiyat osilatörü indikatörünü fiyat değişimi kullanarak çağırdığımızı belirtir. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.PriceOscPoint(10, 30,"Exp");
Sistem.PriceOscPoint(Veriler, 10, 30,"Exp");
```

```
Price ROC Percent - Sistem.PriceRocPercent(12)
```

iDeal indikatör kütüphanesinde yer alan Price Roc Percent isimli indikatörü çağırır. Bir adet paramatre alır ve varsayılan değerleri 12 olarak kullanılır. PERCENT kelimesi, fiyat YÜZDE değişim kullanarak çağırdığımızı belirtir. Belirtilen kadar bar sayısı önceye göre yüzde değişim hesaplar.

Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.PriceRocPercent(10);
Sistem.PriceRocPercent(Veriler, 10);
```

- Price Oscilator Point - Sistem.PriceRocPoint(12)

iDeal indikatör kütüphanesinden yer alan Price Roc Percent isimli indikatörü çağırır. Bir adet paramatre alır ve varsayılan değerleri 12 olarak kullanılır. POINT kelimesi, fiyattaki

nominal (para veya puan olarak) değişim kullanarak çağırdığımızı belirtir. Belirtilen kadar bar sayısı önceye göre parasal değişim hesaplar. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.PriceRocPoint(10);
Sistem.PriceRocPoint(Veriler, 10);
```

- Price Volume Trend - Sistem.PriceVolumeTrend()

iDeal indikatör kütüphanesinde yer alan Price Volume Trend isimli indikatörü çağırır. Hiç paramatre almaz. Sorunsuz ve anlık çalışması için HACİM anlık verisi gerektirir. Aşağıdaki gibi 2 yazım şekli vardır;

```
Sistem.PriceVolumeTrend();
Sistem.PriceVolumeTrend (Veriler);
```

Örnek: İndikatörü çağırıp çizdirmek

var PVT = Sistem.PriceVolumeTrend (); Sistem.Cizgiler[0].Deger = PVT;

- Profit Factor - Sistem. ProfitFactor ()

Bir algoritma geliştirildiği zaman, söz konusu algoritmanın portföyünüzü yönetmesi ve otomatik olarak alım satıma bağlanması öncesi son ve en önemli adım geçmiş dönem performansını ölçmektir. Sistemlerin, stratejilerinizin geçmişten günümüze kullanılıyor olması durumunda ne kadar kazandırdığı/kaybettirdiği çok önemli bir veridir.

PERFORMANS isimli tabloda iDeal kullanıcıları stratejilerinin back-test sonuçlarını/istatistiklerini detaylı olarak görebilir. Bu back test verileri arasında NET GETİRİ ve İŞLEM SAYISI ile birlikte en önemli verilerden biri de PROFIT FACTOR verisidir.

Profit Factor: Kazandıran sinyallerin kazandırdığı paranın, kaybettiren sinyallerin kaybettirdiği paraya oranıdır. Sistemlerin başarını değerlendirmede oldukça önemlidir. Çoğu yatırımcı bu değerin 1.5 üzerinde olmasını istemekle birlikte, hangi oranın daha iyi olduğu elbette sistemin frekansına, trade edilen enstrüman veya piyasanın karakterine ve elbette yatırımcının yorumuna bağlıdır.

Geliştirilen sistemlerin geçmiş dönem getiri hesaplamalarının yapılarak, sistemin profit factor oranının formül içerisinden okutulması ve hatta bu okutulan bilgiye göre stratejinin koşullarının yeniden belirlenmesi mümkündür ve bu nedenle iDeal sistem kütüphanesinde Profit Factor sonuçlarına erişmek mümkündür.

Sistem kütüphanesinden ProfitFactor isimli toplam 4 fonksiyon bulunmaktadır. Bu fonksiyonların kullanım/yazım şekilleri aşağıdaki gibidir;

Sistem.ProfitFactor; //tek bir oran olarak tüm sistemin tüm dönem için Profit Factorü Sistem.ProfitFactorBar(BarSayisi, Kayma); //Girilen bar sayısı kadar öncesine ait PF



Sistem.ProfitFactorIslem(IslemSayisi, Kayma); //Belli işlem sayısı için PF Sistem.ProfitFactorList; //Her bir barda yeniden hesaplanan PF

Geliştirdiğiniz stratejinin getirisini hesaplatıp, örneğin profit factor değeri x'in altında inerse sistemini nakişte geç şeklinde ayaralayabilirsiniz.

```
- Projection Bands (Up/Down) - Sistem.ProjectionUp/Down(14)
```

iDeal indikatör kütüphanesinden yer alan Projection Band isimli indikatörü çağırır. Bir adet paramatre alır ve varsayılan değeri 14 olarak kullanılır. Aşağıdaki gibi yazım şekilleri vardır; (üst ve alt kanalı ayrı ayrı birer fonksiyon ile çağrılır)

```
Sistem.ProjectionDown(14);
Sistem.ProjectionDown(Veriler, 14);
Sistem.ProjectionUp(14);
Sistem.ProjectionUp(Veriler, 14);
```

Örnek: İndikatörü çağırıp çizdirmek

```
var Pro_DOWN = Sistem.ProjectionDown(14);
var Pro_UP = Sistem.ProjectionUp(14);
Sistem.Cizgiler[0].Deger = Pro_DOWN;
Sistem.Cizgiler[1].Deger = Pro_UP;
```

- Projection Oscillator - Sistem.ProjectionOsc(14)

iDeal indikatör kütüphanesinde yer alan Projection Oscillator isimli indikatörü çağırır. Bir adet paramatre alır ve varsayılan değeri 14 olarak kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.ProjectionOsc(14);
Sistem.ProjectionOsc(Veriler, 14);
```

- Projection Bandwith - Sistem.ProjectionBandWith(14)

iDeal indikatör kütüphanesinde yer alan Projection Oscilator isimli indikatörü çağırır. Bir adet paramatre alır ve varsayılan değeri 14 olarak kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.ProjectionBandwidth(14);
Sistem.ProjectionBandwidth(Veriler, 14);
```

- PİVOT - Sistem.PVT01()

iDeal indikatör kütüphanesinde yer alan Pivot İndikatörünü. Çağırır. Parametre olarak pivot değeri hesaplatılmak istenen grafik zaman dilimi (periyodu/dönemi) girilir. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.PVT01(Dönem);
Sistem.PVT01(Veriler, Donem)
```



### Örnek: Haftalık Pivot seviyesinin 5 dklık grafik üzerinde çizdirilmesi;

```
var Pivot = Sistem.PVT01("H");
Sistem.Cizgiler[0].Deger = Pivot;
```



### - Range Indicator - Sistem.RangeIndicator(10,3)

iDeal indikatör kütüphanesinde yer alan Range Indicator isimli indikatörü çağırır. 2 adet paramatre alır ve varsayılan değerler olarak 10/3 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RangeIndicator(10, 3);
Sistem.RangeIndicator(Veriler, 10, 3);
```

### - RAVI - Sistem.RAVI()

iDeal indikatör kütüphanesinde yer alan RAVI isimli indikatörü çağırır. 3 adet paramatre alır ve varsayılan değerler olarak 7/65/Simple kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RAVI(7, 65,"Simple");
Sistem.RAVI(Verileri, 7, 65,"Simple");
```

```
- Relative Momentum Index - Sistem.RelativeMomIndex(20,5)
```

iDeal indikatör kütüphanesinde yer alan Relative Momentum Index isimli indikatörü çağırır. 2 adet paramatre alır ve varsayılan değerler olarak 20/5 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RelativeMomIndex(20, 5);
Sistem.RelativeMomIndex(Liste,20, 5);
Sistem.RelativeMomIndex(Veriler,20, 5);
```

### - Relative Vigor Index - Sistem.RelativeVigorIndex(14)

iDeal indikatör kütüphanesinde yer alan Relative Vigor Index isimli indikatörü çağırır. 1 adet paramatre alır ve varsayılan değerler olarak 14 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RelativeVigorIndex(14);
Sistem.RelativeVigorIndex(Veriler,14);
```

- Relative Vigor Index Signal - Sistem.RelativeVigorIndexSignal(14)

iDeal indikatör kütüphanesinde yer alan Relative Vigor Index Signal isimli indikatörü çağırır. 1 adet paramatre alır ve varsayılan değerler olarak 14 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RelativeVigorIndexSignal(14);
Sistem.RelativeVigorIndexSignal(Veriler, 14);
```

- Relative Volatility Index - Sistem.RelativeVolatilityIndex(14,10)

iDeal indikatör kütüphanesinde yer alan Relative Volatility Index Signal isimli indikatörü çağırır. 2 adet paramatre alır ve varsayılan değerler olarak 14/10 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RelativeVolatilityIndex(14, 10);
Sistem.RelativeVolatilityIndex(Veriler,20, 5);
```

- RSI- Sistem.RSI(14)

iDeal indikatör kütüphanesinde yer alan RSI isimli indikatörü çağırır. 1 adet paramatre alır ve varsayılan değerler olarak 14 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RSI(14);
Sistem.RSI(Liste,14);
Sistem.RSI(Veriler,14);
```

ÖRNEK: RSI indikatörünün kendi ortalamasının üstünde altında olmasıyla al/sat sinyalleri üreten bir sistem;

```
var RSI = Sistem.RSI(40);
var AVR = Sistem.MA(RSI, "Simple", 10);
var SonYon = "";
for (int i=1; i < Sistem.BarSayisi; i++)
{
    if (RSI[i] > AVR[i] && SonYon != "A")
    {
        SonYon = "A";
        Sistem.Yon[i] = "A";
    }
    else if (RSI[i] < AVR[i] && SonYon != "S")
    {
        SonYon = "S";
        Sistem.Yon[i] = "S";
```

### - RSI Denvelope Kanal - Sistem.RSIDenvelope Up/Mid/Down()

iDeal indikatör kütüphanesinde yer alan RSI Denvelope isimli indikatörü çağırır. 3 adet paramatre alır ve varsayılan değerler olarak Simple/14/2 kullanılır. Üst, Alt ve Orta bandı olan bir çeşit kanaldır ve her bir çizgisi ayrı ayrı birer komutla çağırılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RSIDenvelopeDown(Veriler, "Simple", 14, 2);
Sistem.RSIDenvelopeMid(Veriler, "Simple", 14, 2);
Sistem.RSIDenvelopeUp(Veriler, "Simple", 14, 2);
```

### - R Squared - Sistem.RSquared(14)

//ideal

} }

iDeal indikatör kütüphanesinde yer alan R Squared isimli indikatörü çağırır. 1 adet paramatre alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.RSquared(14);
Sistem.RSquared(Veriler,14);
```

#### <u>REF - Sistem.Ref(Liste, Barsayısı)</u>

Bir veri listesinin elemanlarını ileriye veya geriye (sağa veya sola) istenilen eleman (bar) kadar kaydırmak için kullanılır.

ÖRNEK: Kapanış ve Açılış verilerini okuyup, bunları bir bar geriye kaydırmak

```
var Veriler = Sistem.GrafikVerileri;
var C = Sistem.GrafikFiyatOku(Veriler,"Kapanis");
var O = Sistem.GrafikFiyatOku(Veriler,"Acilis");
var RefC = Sistem.Ref(C,-1);
var RefO = Sistem.Ref(0,-1);
```

#### - <u>Renk - Sistem.Renk(Opaklık Oranı, Kırmızı, Yeşil, Mavi)</u>

iDea üzerinden formül, indikatör vs yazılırken, bazen yazı veya çizgiler için renk ataması yapmak gereği doğar. **Sistem.Renk** fonksiyonu, kodlama içerisinde kullanılacak renkleri belirlemek amacıyla kullanılır.

Grafik üzerine çizdirilen çizgiler için bu fonksiyonu kullanmaya gerek yoktur (zorunlu değildir) çünkü çizgi renkleri, panel üzerindeki renk simgelerine tıklayınca açılan renk seçim kutusundan seçilebilir.

Ama grafik zemini veya bar/fiyat değerlerine bir yazı/şekil yazdırılacaksa bunun renginin de belirtilmesi şarttır.

**NOT-1:** Aslında bütün renkler Kırmızı, Yeşil ve Mavi renklerin çeşitli oranlarla karışımından oluşur. IDEAL üzerinde renk tanımları bu mantık üzerine kurulmuştur ve istenen rengin RGB (Kırmızı/Yeşil/Mavi) oranları belirtilmelidir.

**NOT-2:** Bir rengin oranı 0-255 değerleri arasında olabilir. Üç ana renk de sıfır ise SİYAH, 3 ana renk de 255 ise BEYAZ, bu ana renklerden biri 255 diğer ikisi "0" ise, sonuç rengi değeri 255 olan renktir.

**NOT-3:** Windows ortamında renk seçimi yapılabilen herhangi bir pencerede (ideal paneldeki renk kutularında da) renk paleti açılıp herhangi bir renk tonuna kliklendiğinde, ekranda o engin RGB oranları görülebilir.

**NOT-4:** Bu sayfanın en altında bazı popüler renklerin RGB oranları verilmiştir

Sistem.Renk fonksiyonu belirtilmesi gereken 4 adet parametreye ihtiyaç duyar. Bunlar aşağıdadır.

- · Opaklık Oranı (saydamlık oranı)
- · Kırmızı Renk Oranı
- · Yeşil Renk Oranı
- · Mavi Renk Oranı

Sistem.Renk(Opaklık, Kırmızı,Yeşil,Mavi); şeklinde yazılan bu fonksiyonda tüm parametreler girilmek zorundadır.

ÖRNEK: Turkuaz renk (tam saydam): Sistem.Renk(255, 0, 255, 255);

Bazı çok kullanılan renkler;

SARI = Sistem.Renk(255,255,255,1); BEYAZ = Sistem.Renk(255,250,250,250); MAVİ = Sistem.Renk(255,50,100,200); YEŞİL = Sistem.Renk(240,120,255,1); KIRMIZI = Sistem.Renk(255,255,0,0); SİYAH = Sistem.Renk(255,0,0); TURUNCU = Sistem.Renk(255,255,128,0); MOR = Sistem.Renk(255,128,0,128); PEMBE = Sistem.Renk(255,255,0,128); KAHVERENGİ = Sistem.Renk(255,128,0,0); TURKUAZ = Sistem.Renk(255,0,255,255);

Bununla birlikte, renk ataması yapılacağı zaman Color.Red, Color.White şeklindei, renklerin İngilizce isimlerini kullanarak doğrudan bu şekilde de renk ataması yapılabilir.

### - Renk Litesi - Sistem.RenkListesi()

iDeal çizgileri, kendilerine bir renk listesi atanarak çizilebilirler. Bu, her bir barda istenen herhangi bir koşulda çizgiye istediğimiz bir renk atayabilme özgürlüğü demektir.

Her bir bar için içinde renk atanmış bir liste olan Renk Listesini mesela aşağıdaki örmekteki gibi kullanabilirsiniz;

ÖRNEK: Super Trend indikatörü fiyatın üstünde iken başka, altında iken başka renkle çizilsin.

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var ST = Sistem.SuperTrend(3, 10, 10);
var RenkListesi = new List<Color>();
for (int i = 0; i < Sistem.BarSayisi; i++)
    RenkListesi.Add(Color.Gold);
for (int i = 1; i < C.Count; i++)
{
    if (ST[i] > C[i])
        RenkListesi[i] = Color.LightGreen;
    else if (ST[i] < C[i])
        RenkListesi[i] = Color.Red;
}
Sistem.Cizgiler[0].Deger = ST;
Sistem.Cizgiler[0].RenkListesi = RenkListesi;
```



## - Resim Ekle - Sistem.ResimEkle(DosyaAdı, Panel, X, Y)

Grafikler üzerinde çalışırken, ister zevkinize ister sistemlerinizden üretilen sonuçlara göre değişen isterseniz de abone/takipçi/müşteri grubunuza hitaben reklam amaçlı olarak, grafik penceresine fotoğraf/fotoğraflar yerleştirebilirsiniz.

Kodlamanızda Sistem.ResimEkle(DosyaAdi,Panel,X,Y); satırını ekleyerek (bu satırdan istediğiniz kadar yazıp, grafiğin farklı bölgelerine farklı resimler de koyabilirsiniz) bu işlemi yapabilirsiniz.

- **DosyaAdi Parametresi:** çift tırnak için grafiğe eklenecek fotoğrafın bilgisayarınızda bulunduğu yer ve dosyanın uzantısı dahil tam adıdır (Örn: "D:\\manzara2.png")
- **Panel Parametresi:** Fotoğraf, grafiğin hangi bölgesine (kaç numaralı panele) eklenecek bilgisidir. Grafik barlarının çizildiği bölgenin panel numarası SIFIR'dır. Alat kısımdaki indikatör bölgelerine de fotoğraf yerleştirilebilir. Bu bölgelerin panel numarası 1'den başlar ve artarak devam eder.
- **X Parametresi:** Grafiğe yerleştirilecek fotoğrafın, grafiğin en sol bölgesinden (yatay eksen) kaç pixel uzaktan itibaren başlaması istendiği bilgisidir. (sol kenardan 50 pixel uzakta olsun veya sola yapışık olsun (X=0) gibi)
- **Y Parametresi:** Grafiğe yerleştirilecek fotoğrafın, grafiğin en üst bölgesinden (dikey eksen) kaç pixel uzaktan itibaren başlaması istendiği bilgisidir. (en üst kenardan 50 pixel uzakta olsun veya üste yapışık olsun (Y=0) gibi

IDEAL üzerinden yapıp çeşitli mecralarda paylaştığınız analiz ve görsellerinizi kendi logonuzla da paylaşmanızı kolaylaştırır

ÖRNEK://SİSTEMİM AL pozisyonundaysa BOĞA fotosu, SAT pozisyonundaysa AYI resmi, diğer durumlarda (FLAT ise) manzara fotosu yerleştiren kod örneğği.

```
if (SonYon == "A")
   Sistem.ResimEkle("D:\\boga.png",1,0,40);
else if ((SonYon == "S")
   Sistem.ResimEkle("D:\\ayi.png",1,0,40);
else
   Sistem.ResimEkle("D:\\manzara.png",1,0,40);
```

NOT-1: Sistemin yönü değiştiği anda, arka plan fotoğrafı hemen değişir.

NOT-2: Fotoğraf grafik çizgilerinin arka planına gönderilir. Bu sayede grafik bir manzaranın/logonun üzerine çizilmiş gibi olur. NOT-1: Sistem.ResimEkle satırından birden fazla yazılarak, grafiğe istenildiği kadar fotoğraf/logo eklenebilir.



### <u>Robot Fonksiyonları - Sistem.Robotxxxxx</u>

iDealde en çok kullanılan fonksiyonlar arasında tek satırlık robot fonksiyonları yer alır. Kullanıcılar tarafından yazılmış ve grafikler üzerinde çalıştırıldıklarında AL/SAT sinyallri üreten sistemler/stratejiler bu tek satırlık komutlarla kolayca robota bağlabilirler.

Toplam 5 adet tek satırlık robot fonksiyonu bulunmaktadır. Bunlar aşağıdaki şekilde çağırılıp kullanırlar;

Not: Aktif kelimesi, tüm emirlerin aktiften (PİYASA EMRİ) gönderdiğini vurgular.

Sistem.RobotHisseAktifAcigaVar(SistemAdi, BazSembol, EmirSembol, Periyot, Miktar); Sistem.RobotHisseAktifAcigaYok(SistemAdi, BazSembol, EmirSembol, Periyot, Miktar); Sistem.RobotViopAktif(SistemAdi, BazSembol, EmirSembol, Periyot, Miktar); Sistem.RobotViopTumGun(SistemAdi, BazSembol, EmirSembol, Periyot, Miktar); Sistem.RobotViopGunSonuKapat(SistemAdi, EmirSembol); Sistem.RobotStop();

Kural şudur; Komutların içindeki paramatreler girilir, ve tek bir satırdan oluşan bu kodlar "Islem" isimli bir değişkene atanıp kaydedilir. Kaydedilen bu kod artık bir robottur ve sanal veya gerçek aksiyonda çalıştırılabilir. Komutlar ve içlerindeki bilgi girişleri aşağıda açıklanmıştır;

### <u>RobotHisseAktifAcigaVar : Hisse senetlerinde çalışır ve Açığa Satış yapar</u> (stratejide açığa sat koşulu var ise)

<u>SistemAdi</u> = al/sat sinyalleri üreten stratejinizi kaydettiğiniz formülün ismi (örmek: MA1) <u>BazSembol</u> = Sinyaller hangi sembolün grtafikleri üzerinden hesaplanıyor (Örnek: "IMKBH'GARAN) <u>EmirSembol</u>: Al veya Sat sinyali gelince alınacak veya satılacak sembol (Örnek: "IMKBH'GARAN") <u>Periyot:</u> Sinyaller hangi zaman periyotlu grafiklerden geliyor (örnek 5 dk için "5")

Miktar: Al veya Sat gelince kaç lot emir iletilecek (örnek 100)

Örnek Tek Satır Robot: var Islem = Sistem.RobotHisseAktifAcigaVar("MA1", "IMKBH'GARAN", "IMKBH'GARAN", "5", 100);

#### <u>RobotHisseAktifAcigaYok : Hisse senetlerinde çalışır ve Açığa Satış yapmaz. Sadece</u> <u>Alım yapılır ve sat gelince eldeki lot satılır.)</u>

<u>SistemAdi</u> = al/sat sinyalleri üreten stratejinizi kaydettiğiniz formülün ismi (örmek: MA1) <u>BazSembol</u> = Sinyaller hangi sembolün grtafikleri üzerinden hesaplanıyor (Örnek: "IMKBH'GARAN) <u>EmirSembol</u>: Al veya Sat sinyali gelince alınacak veya satılacak sembol (Örnek: "IMKBH'GARAN") <u>Periyot:</u> Sinyaller hangi zaman periyotlu grafiklerden geliyor (örnek 5 dk için "5") <u>Miktar</u>: Al veya Sat gelince kaç lot emir iletilecek (örnek 1)

Örnek Tek Satır Robot: var Islem = Sistem. RobotHisseAktifAcigaYok("MA1", "IMKBH'GARAN", "IMKBH'GARAN", "5", 100);

### RobotViopAktif : VIOP kontratları için çalışır ve Piyasa Emri iletir)

<u>SistemAdi</u> = al/sat sinyalleri üreten stratejinizi kaydettiğiniz formülün ismi (örmek: MA1) <u>BazSembol</u> = Sinyaller hangi sembolün grtafikleri üzerinden hesaplanıyor (Örnek: "VIP'VIP-X030) <u>EmirSembol</u>: Al veya Sat sinyali gelince alınacak veya satılacak sembol (Örnek: "VIP'F\_XU0301221") <u>Periyot:</u> Sinyaller hangi zaman periyotlu grafiklerden geliyor (örnek 5 dk için "5") <u>Miktar</u>: Al veya Sat gelince kaç lot emir iletilecek (örnek 1)

Örnek Tek Satır Robot: var Islem = Sistem.RobotViopAktif("MA1", "VIP'VIP-X030", "VIP'F\_XU0300221", "5", 1);

#### <u>RobotViopAktifTumGun : VIOP kontratları için çalışır, Piyasa Emri iletir ve Akşam</u> <u>Seansı dahil çalışır)</u>

<u>SistemAdi</u> = al/sat sinyalleri üreten stratejinizi kaydettiğiniz formülün ismi (örmek: MA1) <u>BazSembol</u> = Sinyaller hangi sembolün grtafikleri üzerinden hesaplanıyor (Örnek: "VIP'VIP-X030) <u>EmirSembol</u>: Al veya Sat sinyali gelince alınacak veya satılacak sembol (Örnek: "VIP'F\_XU0301221") <u>Periyot:</u> Sinyaller hangi zaman periyotlu grafiklerden geliyor (örnek 5 dk için "5") <u>Miktar</u>: Al veya Sat gelince kaç lot emir iletilecek (örnek 1)

Örnek Tek Satır Robot: var Islem = Sistem.RobotViopAktifTumGun("MA1", "VIP'VIP-X030", "VIP'F\_XU0300221", "5", 1);

Not: Akşam seansında piyasa emri göndermek yasaktır. Bu seans diliminde AL sinyali gelirse emirler LİMİTLİ olarak TAVAN FİYATA, SAT sinyali gelirse emirler LİMİTLİ olarak TABAN FİYATA gönderilir.

<u>RobotViopGunSonuKapat : VIOP kontratları için çalışır,Piyasa Emri iletir ve günün</u> <u>sonunda varsa pozisyonu kapatır. (ertesi gün sisteminiz hangi yönde ise o yönde</u> <u>tekrar pozisyon açşlılır.)</u>

<u>SistemAdi</u> = al/sat sinyalleri üreten stratejinizi kaydettiğiniz formülün ismi (örmek: MA1) <u>EmirSembol</u>: Al veya Sat sinyali gelince alınacak veya satılacak sembol (Örnek: "VIP'F\_XU0301221")

Örnek Tek Satır Robot: var Islem = Sistem.RobotViopGunSonuKapat(SistemAdi, EmirSembol);

RobotStop : Çalışan bütün robotları durdurmak amaçlı kullanılır

Herhangi bir koşulda çalışan BÜTÜN robotları durdurmak amaçlı kullanılır. Örneğin endenksin değer kaybı %5 olursa robotlar kapatılsın şeklinde bir kod olarak kaydedip bu kodu da robot olarak çalıştırabiliriz.


```
var Endeks = Sistem.YuzeyselVeriOku("IMKBX'XU100");
var EndeksDegisim = Endeks.NetPerDay;
if (EndeksDegisim < -5 )
{
   Sistem.RobotStop();
}
```

- Saat - Sistem.Saat()

iDeal Yayın saatine erişmek için kullanılan sistem fonksiyonudur. Saat/Dakika/Saniye olarak iDeal yayınındaki saati veririr.

Örnek kullanım: ekrana o anki saati mesaj olarak getiren kod:

```
var saat = Sistem.Saat;
Sistem.Mesaj(saat);
```

Saat verisi kullanılarak yapılan çok fazla kodlama vardır ve sıkça kullanılır. Burada dikkat edilmesi gereken en önemli husus, bu komutla alınan saatin, KODUN ÇALIŞTIĞI ANDAKİ saat olduğudur. Grafik barlarının saati değildir. GRAFİK VERİLERİ komutu anlatımında, barların saat verisine nasıl erişilebileceği anlatılmıştır.

Belli saatlerde belli işlemleri yaptırmak veya robotlara belli saatlerde çalışma gibi komutlar vermek için saat bilgisi kullanıcı tarafından tanımlanan saatlerle kıyaslanır.

Eğer saat xx:xx:xx den büyükse pozisyonu kapat

Eğer saat xx:xx:xx den küçükse AL gibi.

Zaman verileri ile ilgili KISYASLAMA için c# yazım dilinde CompareTo komutu kullanılır. Compare İngilizce bir klime olarak KIYASLA anlamındadır.

CompareTo(x) şeklinde bir kullanım, verilen zamanı x ile kıyaslar ve büyüktür sıfır, küçüktür sıfır, eşittir sıfır, büyükeşit, küçükeşittir sıfır gibi sonuçlar verir.

ÖRNEK: Sistemin saati kullanıcı tarafından girilen bir saatten (11:05:30) büyük veya eşit olduğu anda 1 lot DOHOL al kodu

```
var Sembol = "IMKBH'DOHOL";
var Miktar = 1;
var islemsaati = "11:05:30";
if (Sistem.Saat.CompareTo(islemsaati) >= 0)
{
    Sistem.EmirSembol = Sembol;
    Sistem.EmirIslem = "Alış";
    Sistem.EmirIpi = "Piyasa";
    Sistem.EmirSuresi = "KIE";
    Sistem.EmirSuresi = "KIE";
    Sistem.EmirMiktari = Miktar;
    Sistem.EmirGonder();
}
```

ÖRNEK 2: Grafik barlarının saatlerini kullanma amaçlı olarak, RSI 55 ten büyükse AL yapan, RSI 45 ten küçükse SAT yapan, saat 17:40 barında nakite geçen sistem;

```
var V = Sistem.GrafikVerileri;
var RSI = Sistem.RSI(14);
var SonYon="";
for (int i = 1; i < V.Count; i++)</pre>
{
   if (V[i].Date.Hour >= 17 && V[i].Date.Minute >= 40 && SonYon != "F")
   {
      Sistem.Yon[i] = "F";
      SonYon = "F";
   }
   else if (RSI[i] > 55 && SonYon != "A" )// % 2 stop
   {
      Sistem.Yon[i] = "A";
      SonYon = "A";
   }
   else if (RSI[i] < 45 && SonYon != "S" )// % 2 stop</pre>
   {
      Sistem.Yon[i] = "S";
      SonYon = "S";
   }
}
```

NOT: Barlara ait saat / tarih bilgilerine erişim için ipucu kodları:

```
var V = Sistem.GrafikVerileri;
for (int i = 1; i < V.Count; i++)</pre>
{
   if (V[i].Date.DayOfWeek.ToString() == "Monday" )
      Sistem.Yon[i] = "A";
   else if (V[i].Date.DayOfWeek.ToString() == "Friday")
      Sistem.Yon[i] = "S"
   else if (V[i].Date.ToString("yyyyMMdd")=="20171204")
      Sistem.Yon[i] = "A";
   else if (V[i].Date.Hour == 18 && V[i].Date.Minute == 05)
      Sistem.Yon[i] = "F";
   else if (V[i].Date.Month == 12)
      Sistem.Yon[i] = "F";
   else if (V[i].Date.Year == 2016)
      Sistem.Yon[i] = "F";
}
```

- Saat Araliiği - Sistem.SaatAraligi("15:00", "16:30")

Belli saatler arasında bir iş yaptırmak veya yaptırmamak için Saat Aralığı fonksiyonu kullanuılabilir.

ÖRNEK: Saat 09:30 ile 09:50 arasında ve 17:50 ile 18:00 arasında hiçbir işlem yapma, geri kalan zamanlarda formülü çalıştır;

```
if (Sistem.SaatAraligi("09:30", "09:50") || Sistem.SaatAraligi("17:50", "18:00"))
{
     }
else
{
     // calismalariniz
}
```

### - Satış Fiyat - Sistem.SatisFiyat(Sembol)

Bir sembolün o an ki **en iyi satış fiyatını** okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) en iyi satış fiyatı okutulmak istenen sembol yazılır.

Not: iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir. Örneğin USDTYR için FX'USDTRY şeklinde yazılır. Bir sembolün PİYASA kodunun ne olduğu, o sembolü sayfanıza yazarken @ işareti yanında gösterilir.)

Örnek kullanım şekli aşağıdaki gibidir.

```
var Sembol = "IMKBH'GARAN";
var X = Sistem.SatisFiyat(Sembol);
Sistem.Mesaj(X.ToString());
```

<u>Satış Lot - Sistem. SatisLot(Sembol)</u>

Bir sembolün o an ki **en iyi satış fiyatında bekleyen lot miktarını** okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) en iyi satış fiyatında bekleyen lot miktarı okutulmak istenen sembol yazılır.

```
var Sembol = "IMKBH'GARAN";
var X = Sistem.SatisLot(Sembol);
Sistem.Mesaj(X.ToString());
```

🛎 🗧 GAR.	AN.Y I	о скта	вкн	-			- *
Günlük		ot 123,68	6,592	Ortalama	9.8063		
5 Kurum	I PO	GÇ 28,23	4,899	UNS	IYM		
Tavan 10.85 Taban 8.89		Yks 9.88 Dşk 9.71	Önci 9.77 Lot 123,686	,589	Frk% 1.02 Frk 0.10	Dng.Fy 9.87 Aort 9.8070	rt I
A.Saat	A.E	A.Lot	Alış	Satış 👔	JILUU	S.E	S.Saat
18:09:24 18:08:00 18:09:35 18:09:05 18:05:22 18:08:00 18:08:00 18:05:2	4 9 61 72 62 54 99	4,783 86,175 128,916 435,000 389,615 289,279 370,735	9.86 9.85 9.84 9.83 9.82 9.81 9.80	9.87 9.88 9.89 9.90 9.91 9.92 9.93	411,395 526 027 488,160 946,163 319,606 184,917 117,660	38 171 128 310 77 80 50	18:10:00 18:08:00 18:08:53 18:09:44 18:05:22 18:05:22
	11395	5					
1							

### - Sayı Listesi - Sistem.SayiListesi()

iDeal sistem modülünde formüller yazarken çokça kez LİSTE tanımları kullanılır. Başlangıçta tüm elemanları boş veya sıfır olan listeler tanımlar, sonra kendi koşullarımıza göre bu listelerin elemanlarını doldurur ve bunları genelde çizgi olarak çizer veya başka indikatörleri giriş listesi olarak veriririz. İleriki sayfalarda anlatılacak olan SİSTEM GETİR fonksiyonu, yazılmış bir formülü çağırmak ve o formül içindeki bazı bilgileri kullanmak amaçlı bir komuttur. SİSTEM GETİR ile xxx isimli bir sistemi çağırdığımızda, o sistem içindeki bir hesaplanmış listeye erişmek ihtiyacı varsa o zaman listeyi SAYI LİSTESİ olarak tanımlamış olmak gerekir.

Yani bir sistemde x ve y isimli 2 liste olsun; Bu sistemi SİSTEM GETİR ile çağırdığımızda x listesine erişebiliriz SAYI LİSTESİ olarak tanımlandığı için.

```
var x = Sistem.SayiListesi;
var y = Sistem.Liste(0);
for (int i = 1; i<Sistem.BarSayisi; i++)
{
    if (A KOŞULU))
        x[i] = C[i];
    else if (B KOŞULU)
        y[i] = H[i];
}
```

### - Sayı Tablosu Kontrol Oku/Güncelle - Sistem.SayiTablosunu Oku/Guncelle()

iDeal veri terminali üzerinde robotlar çalıştırıken çoğu zaman bazı bilgileri bir dosyada veya bellekte saklamak ve gerektiğinde kullanmak gerekir. Kod yazılırken bilgi atanan (var x = 5; şeklinde tanımlanan) değişkenler bazen bu ihtiyacı karşılamaz.

Çünkü kod içerisinde tanımlanan değişkenler, kodun her seferinde dafalarca çalışması durumunda hep ilk tanımladığı değerine döner ve ilerleyen satırlarda hesaplamalar sonucu yeni değeri bulunur. Ama bu döngü her seferinde yeniden olur. Oysa istenen şey, bir kez değeri değiştirilen bir verinin, bir daha kullanıcının tanımladığı bir koşul olmadan hiç değişmemesidir. Bu nedenle iDealde Pozisyon Kontrol, Sayı Tablosu, Sözcük Tablosu gibi veri saklama işlevi olan komutlar kullanılır.

Sayı Tablosu, sayısal verilerin saklandığı bir bellek değişkenidir. Kod içerisinde sadece istenen koşullar gerçekleştiği zaman değeri değiştirilir. Kod her çalşıştığında yeniden başlangıuçta tanımlanan değeri almaz.

Sayı Tablosunda saklanan bilgiler RAM de durur ve iDeal kapanırsa bu bilgiler sıfırlanır.

ÖRNEK: Bir işlemin GÜNDE 1 DEFA yapılması isteniyor. Bu durumda günün tarihi bir anahtar olarak SayiTablosu komutuna verilir. Gün değişmemişse bir daha işlem yapmaması sağlanır.

```
var Anahtar = Sistem.Name + ", "
Sistem.GrafikVerileri[Sistem.GrafikVerileri.Count-1].Date.ToString("yyyyMMdd");
var Defa = Sistem.SayiTablosunuOku(Anahtar);
if (Defa == 0)
{
```



}

Sistem.SayiTablosunuGuncelle(Anahtar, 1);

```
/// kodunuzun tamamı burada
```

```
- Sayı Yuvarla - Sistem.SayiYuvarla(Sayı, Adım)
```

Formül yazarken yapılan hesaplamalarda elde edilen çeşitli sayılar/fiyatlar küsuratlı olabilir. Özellikle sembolün fiyat adımlarına göre yapılacak kıyaslarda, bu sayıların o sembolün işlem görebileceği bir fiyat seviyesinde olabilmesi için yuvarlanması ihtiyacı doğabilir.

Sayı Yuvarla fonksiyonu, herhangi bir sayısı, kullanıcı tarafından tanımlanan formatta ve adımda bir sayıya yuvarlamak için kullanılır

```
//2.547 sayısını, 0.005'erlik adımlara uygun olacak şekilde yuvarlamak
```

```
var sayi = 2.547;
var x = Sistem.SayiYuvarla(sayi, 0.005);
Sistem.Mesaj(x.ToString());
```

```
var sayi = 2.547;
var x = Sistem.SayiYuvarla(sayi, 0.005);
Sistem.Mesaj(x.ToString());
iDeal Mesaj 09.10.2015 12:31:00
```

```
2.545
```

/VIOP sözleşmesinin önceki günkü High ve Low değerlerinin toplamının yarısı olan sayısı .547 sayısını, 0.005'erlik adımlara uygun olacak şekilde yuvarlamak

```
var Sembol = "VIP'VIP-X030";
var Veri = Sistem.YuzeyselVeriOku(Sembol);
var UZL = Veri.SettlementPrice;
var SonFiyat = Sistem.SonFiyat(Sembol);
float Fark = UZL - SonFiyat;
Fark = Sistem.SayiYuvarla(Fark, 0.025);
Sistem.Mesaj("Son Fiyat ile Uzlaşı arasındaki fark " + Fark + " puan");
```

Yuvarlama yapılmaz ise sonuç;



- Select BarNo (Mouse ile Kliklenen Barı Okumak) - Sistem.SelectBarNo

iDeal kullanıcılarıi grafik üzerinde Mouse ile bir noktaya kliklendiği zaman, o kliklenen barın tarihini, saatini, bar numarasını sistem üzerinden okuyabilirler. Bu sayede mesela Mouse ile tıkladıkları noktadan geriye doğru olan bölgedeki yükselen veya düşen trend çizimlerini otomatik olarak ve sadece Mouse ile tıklayarak yaptırabilirler. Aynı şekilde pivot seviyeleri çizimini de Mouse tıklaması ile istedikleri bölge için kolayca yapabilirler.

#### Örnek1: Kliklenen barın tarih ve saatini öğren;

```
var V = Sistem.GrafikVerileri;
int nokta = Sistem.SelectBarNo;
Sistem.Mesaj(V[nokta].Date.ToString("yyyyMMdd HH:mm"));
```

//	DC	ΔL	
Ska	la Periyot		
Tarih Açılış Yüksek Düşük Kapanış Hacim Lot Ort Frk	18.12.2020 13:25 1561.00 1551.50 1561.25 1561.25 1,233,355 79 156.12 0.25 (% 0.02)		
Selecte	edBar		
\$020	1218 13:25		

ÖRNEK2: Mouse ile tıklanan bardan günümüze kadar olan bölgede varsa yükselen ve düşen trendleri bul ve çiz:

```
int donem = Sistem.BarSayisi - Sistem.SelectBarNo;
Sistem.Cizgiler[0].Deger = Sistem.OtoTrendYukselen(donem, 10);
Sistem.Cizgiler[1].Deger = Sistem.OtoTrendDusen(donem, 10);
```



- Select Tarih(Mouse ile Kliklenen Barın TarihiniOkumak) - Sistem.SelectTarih

iDeal kullanıcıları grafik üzerinde Mouse ile bir noktaya kliklediği zaman, o kliklenen barın tarihini ve saat olarak tam zamanını sistem üzerinden okuyabilirler.

#### ÖRNEK: Mouse ile kliklenen barın tam tarih + saat bilgisi:

<pre>var tarih = Sistem.Selec Sistem.Mesaj(tarih.ToStr</pre>	r tarih = Sistem.SelectTarih; stem.Mesaj(tarih.ToString()); ✓ VIP-X030				
<ul> <li>✓ 1/1P-X030</li> <li>↓ ORJ</li> <li>1 dk</li> <li>+ I</li> <li>T</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y</li> <li>Y<!--</td--></li></ul>					
✓ VIP-X030					
15					
Tarih         15.12.2020         12:33           Açılış         1541.50           Yüksek         1542.00           Düsük         1540.75					

TAYDKHLOF	arih Açılış Yüksek Düşük Capanış Lacim Ot Ot Ort İrk	15.12.2020 12:33 1541.50 1542.00 1540.75 9.278,875 602 154.13 -0.50 (% -0.03)		water of the second -	----	--
		_Y				
	15.12	.2020 12:33:	00			

### - Sembol - Sistem.Sembol()

Üzerinde işlem yapılan, grafiği ,üzerinde bir takım çizim veya stratejiler uygulanan sembolün (kodun) bilgisi bu fonksiyonla okutulur. IDEAL programında BÜTÜN kodlar işlem gördükleri piyasada kullanıldıkları kısa kod ve dahil oldukları PİYASA'nın koduyla birlikte tanımlıdır.

Örneğin BIST Hisse senetleri IMKBH, BIST endeksleri IMKBX, Serbest Piyasa Döviz/Altın kodları SERPIY, Uluslararası spot pariteler FX, Vadeli işlem piyasası kontratları VIP, Tahvil Bono sözleşmeleri THVX kısa market kodlarına sahiptir.

IDEAL sayfasında bir boş hücreye bir kod klavye girişi ile yazılmaya başlandığında, program size yazdığınız harf kadarıyla uyuşan kodları bir liste olarak sunar ve o listede PİYASA KODU da gösterilir

ÖRNEK: GARAN hisse kodunun son fiyatının, HALKB hisse senedi son fiyatına oranını bulan (bu oran x olunca bir işlem yapan bir kod olarak kullanım devam edebilir) kod örneği

```
var Sembol1 = "IMKBH'GARAN";
var Sembol2 = "IMKBH'HALKB";
```

var GARAN\_Fiyat = Sistem.SonFiyat(Sembol1); var HALKB\_Fiyat = Sistem.SonFiyat(Sembol2);

var Oran = GARAN\_Fiyat / HALKB\_Fiyat ;

Sistem.Mesaj(Oran.ToString());

🖳 iDeal Mesaj 07.10.2015 09:34:52

0.6995305

### - Sembol Adları Listesi - Sistem.SembolAdListesi(market, seri)

Herhangi bir ihtiyaç için tüm sembol listesini veya belli bir grup sembolün listesini elde etmek istediğimizde bu komutu kullanabiliriz.

Daha önceki fonksiyon tanıtımlarında da değindiğimiz gibi iDeal programında bir sembolün tam yazılışı dahil olduğu piyasa ile birlikte olur. Örneğin GARANTİ BANKASI HİSSE SENEDİ kodunun tam yazılışı IMKBH'GARAN şeklindedir. Buradaki IMKBH kısmı prefix (ön ek) tir ve dahil olduğu piyasayı belirtir (IMKBH = İMKB Hisse)

Hisse piyasasında hisselerin seri kodları vardır. E (Eski), R (Rüçhan), V (Varant), F (Borsa Yatırım Fonu), TE (Temerrüt), BE (Birincil Pazar), HE (Jalkha arz) gibi.

Örneğin bütün Rüçhan Hakkı Pazarı sesisi olan sembolleri erişmek istersek aşağıdaki kodu yazabiliri;

```
var Liste = Sistem.SembolAdListesi("IMKB", "R");
var Metin = "";
foreach(var item in Liste)
{
    Metin += item + "\r\n";
}
Sistem.Mesaj(Metin);
```

#### HisseSembolListesi

IMKBH' ADESER IMKBH' AFYONR IMKBH' AGYOR IMKBH' AKBNKR IMKBH' ALBRKR IMKBH' ATLASR IMKBH' AYCESR IMKBH' AYCESR IMKBH' AYENR IMKBH' BNTASR IMKBH' BOSSAR IMKBH' BOYPR



Aynı formülde Eski serisi olan (hisse senetleri) odlara erişim için R harfini E yaparsak sonuç listesi aşağıdaki gibi olur;

	HisseSembolListesi
	IMKBH'BFREN
	IMKBH'BIMAS
	IMKBH'BIZIM
l	IMKBH'BJKAS
l	IMKBH'BLCYT
ł	IMKBH'BMEKS
	IMKBH'BMELK
	IMKBH'BNTAS
	IMKBH'BOLUC
	IMKBH'BOSSA
	IMKBH'BOYP
	IMKBH'BRISA

VIOP işlem gören vadeli kontratların YAKIN VADE kod listesini elde etmek istersek;

```
var Liste = Sistem.SembolAdListesi("VIP'VIP-", "");
var Metin = "";
foreach(var item in Liste)
{
    Metin += item + "\r\n";
}
Sistem.Mesaj(Metin);
```

AktifVadeliVIOPSembolListesi

```
VIP'VIP-KOZAL
VIP'VIP-KRDMD
VIP'VIP-MGROS
VIP'VIP-OYAKC
VIP'VIP-PETKM
VIP'VIP-PGSUS
VIP'VIP-RUB
VIP'VIP-SAHOL
VIP'VIP-SASA
VIP'VIP-SISE
VIP'VIP-XSIN-T
VIP'VIP-SOKM
```

- Sembol Veri Listesi - Sistem.SembolVeriListesi(market, seri)

Herhangi bir ihtiyaç için tüm sembol listesine ait bir veya birkaç veriyi direk okumak için kullanılır.

ÖRNEK: Bütün hisselerin yüzde değişimlerini oku ve ekrana göster;

```
SEZAİ KILIÇ
```

```
var Liste = Sistem.SembolVeriListesi("IMKB", "E");
double Hacim = 0;
float OrtalamaGetiri =0;
var Metin = "";
foreach(var item in Liste)
{
    Metin += item.Root + " % " + item.NetPerSession.ToString("0.00" ) + "\r\n";
}
```

Sistem.Mesaj(Metin);

SembolListeleri2					
BAYRK	% 4.53				
BERA	% -4.16				
BEYAZ	% −1.03				
BFREN	% 1.40				
BIMAS	% 1.90				
BIZIM	% 0.26				
BJKAS	% 0.49				
BLCYT	% 1.90				
BMEKS	% 0.00				
BMELK	% 0.00				
BNTAS	% 0.22				
BOLUC	% 0.00				

### - <u>Sembol Tanımla - Sistem.SembolTanimla(SembolAdı, OndalıkBasamakSayısı)</u>

iDeal Sistem modülünü kullanarak, kendi sembol, endeks veya sepetlerini oluşturabilirsiniz. Kendinize özel, çeşitli formüller içeren, başka sembollerin verilerini de kullanarak yeni bir veri üreten sembol ya da semboller tanımladığınızda ve bu sembollerin fiyat veya diğer yüzeysel bilgilerinin sürekli olarak güncellenmesini istediğiniz, iDeal'in Kullanıcı Sembol Sistemleri özelliğini kullanmanız gerekmektedir

Bu özellik, sembol tanımlamalarının tamamının **KullaniciSembolleri** ismiyle kaydedilmesi ve ideal ana menüsü altındaki **Özellikler** satırından **Kullanıcı Sembollerini Aktifleştir** seçeneğini işaretlemenizle çalışır.

Sembol Tanımla fonksiyonunu **Sistem.SembolTanimla("sembol adı", basamak sayısı);** satırını (satırlarını) ekleyerek kullanabilirsiniz. Bu satır eklenip, bir kez formül test denildiği anda sembol yaratılır ve iDeal'in semboller listesine de eklenir.

- Sembol tanımı satırı tek başına çok bir anlam ifade etmez. Bu yaratılan sembole veri gelmesini de sağlamak için gerekli (sembolün verisine kaynak olacak) formülleri de yazmak gerekir.
- Fonksiyonu kullanırken, parantezin içindeki ilk parametre SEMBOL ADI parametresidir.
   Çift tırnak içinde, yaratılacak sembolün yer alacağı piyasa ve (üstten tek tırnakla ayırarak) sembole vereceğiniz KISA KOD birlikte sembol adını oluşturur. (Örnek: "DFN'TEST123" veya "IMKBH'GRNHLB" veya "SERPIY'GUMUSERUO" gibi.)

- Yaratılan sembolleri genelde DFN prefixi ile ("DFN'SEMBOL1") tanımlamanızı öneririz. Bu market koduyla oluşturulan ve DirectFN tarafından da bu market koduyla yayını sağlanan tüm kodlar, menüde ÇEŞİTLİ PİYASALAR altında görülebilir.
- Fonksiyonun ikinci parametresi, yaratılan sembolün fiyat bilgisinin, noktadan sonra kaç basamak olacağının belirtilmesidir.
- Sembol tanımlandıktan sonra, sembole ait YÜZEYSEL BİLGİ ALANLARI da atanabilir. (uzun adı, son fiyatı, önceki gün kapanışı vs)

#### **ÖRNEK:**

- var DENEMEKODU1 = Sistem.SembolTanimla("DFN'DENEMEKODU1", 2);
- DENEMEKODU1.Description = "Test Amacli Deneme Kodudur";
- DENEMEKODU1.LastPrice = GARAN.LastPrice / HALKB.LastPrice ;

SembolTanimla fonksiyonu, tek başına kullanıldığında sadece sembolü yaratır. Sistem kodunda bu yaratılan sembolün fiyatının ne olacağına dair formül olsa bile, değer sadece bir kez (kod kaydedildiğinde veya formül test yapıldığında) oluşur/değişir. Girilen formüle uygun olarak, sembolün değerinin de sürekli değişmesini sağlamak için ayrı, gelen her değerin saklanması ve bu sembole ait bir grafik veri bankası oluşması için ayrı birer yardımcı fonksiyon daha vardır

- Sistem.YuzeyselGuncelle(Sembol): Her 1 saniyede bir formülü/kodu çalıştırır ve yaratılan sembolün, kullanıcı tarafından atanmış tüm yüzeysel bilgi alanlarını günceller ve güncelleme olduğunda ekranda update rengi (yanıp sönme) değişir (SON FİYAT, ALIŞ FİYATI, SATIŞ FİYATI, TANIM vs).
- Sistem.GrafikGuncelle(Sembol): Tanımlanmış sembole ait SONFİYAT bilgisi her güncellendiğinde, bu sembolün kendisi için oluşturulan dosyalarda GRAFİK VERİSİ saklanmaya başlar ve yaratılan kodun kendi grafiği üzerinden teknik analiz yapılabilir.

### ÖRNEK KODLAR:

```
var SGLDD = Sistem.YuzeyselVeriOku("SERPIY'SGLDD");
var SUSD = Sistem.YuzeyselVeriOku("SERPIY'SUSD");
//CEYREK ALTIN TANIMLA
var CEYREK = Sistem.SembolTanimla("DFN'CEYREK", 3);
CEYREK.Description = "Çeyrek Altın";
CEYREK.BidPrice = Convert.ToSingle(((SGLDD.BidPrice-10)*SUSD.BidPrice/0.995)/1000*1.58);
CEYREK.AskPrice = Convert.ToSingle(((SGLDD.AskPrice+10)*SUSD.AskPrice/0.995)/1000*1.67);
CEYREK.LastPrice = (CEYREK.BidPrice+CEYREK.AskPrice)/2;
Sistem.YuzeyselGuncelle(CEYREK);
```

```
// ATA ALTIN TANIMLA
var ATA = Sistem.SembolTanimla("DFN'ATA", 2);
ATA.Description = "Yarım Altın";
ATA.BidPrice = Convert.ToSingle(((SGLDD.BidPrice-10)*SUSD.BidPrice/0.995)/1000*3.18);
ATA.AskPrice = Convert.ToSingle(((SGLDD.AskPrice+10)*SUSD.AskPrice/0.995)/1000*3.35);
ATA.LastPrice = (ATA.BidPrice+ATA.AskPrice)/2;
Sistem.YuzeyselGuncelle(ATA);
```



🗶 🍝 Fiyat Penceresi	TRK YD PYS Para			
Kod	Tanım	Son.Fyt	AI.Fyt	Sat.Fyt
CEYREK	Çeyrek Altın	179.138	174.052	184.225
ATA	Yarım Altın	744.65	741.37	747.93

<u>Sembol İşlemlerini Oku - Sistem.SembolslemleriniOku(Sembol, Tarih)</u>

iDeal Sistem modülünü kullanarak, bir hisse veya vadeli sözleşmenin herhangti bir güne ait gerçekleşen işlemleri (zaman satış tablosu) okunabilir. Bu komut ile okunan işlemler listesinde her bir işlem için saat, fiyat, lot, (lisans varsa) alan ve satan kurum bilgileri yer alır.

Okuma komutu hisse ve viop için aşağıdaki gibidir;

```
var IslemlerVIOP = Sistem.SembolIslemleriniOku("VIP'F_XU0301220", "10/12/2020");
var IslemlerHisse = Sistem.SembolIslemleriniOku("IMKBH'GARAN", "17/12/2020");
```

Örnek olarak bir günün GARAN işlemlerini okuyup bir aracı kurumun (mesela Teb) için NET LOT bilgisi hesaplatmak.

```
var IslemlerHisse = Sistem.SembolIslemleriniOku("IMKBH'GARAN", "17/12/2020");
var NetLot = 0;
foreach (var item in IslemlerHisse)
{
    if (item.BuyerCode == "TBY") NetLot += item.Size; //alışları ekle
    if (item.SellerCode == "TBY") NetLot -= item.Size; //satışları çıkar
}
Sistem.Mesaj(NetLot.ToString("0,000"));
```

- Ses Çalma - Sistem.Ses()

İstenilen bir anda veya bir koşula bağlı olarak, bir ses dosyasının oynatılması için kullanılır. (Örnek ABCD hissesinin fiyatı önceden girilmiş bir değeri görürse zil sesi çalsın)

<u>ÖRNEK:</u>//EREGL hissesinin Son İşlem Fiyatı 3.84'e eşit veya büyük olunca Media klasöründeki "dingdong" isimli ses dosyasını çal

```
var Sembol = "IMKBH'EREGL";
var SembolDeger = Sistem.SonFiyat(Sembol);
if (SembolDeger >= 3.84f)
  Sistem.Ses("Media\\dingdong.wav");
```

Not: Fonksiyon Dosya Adı verilmeden kullanılırsa, windows'un klasik "ding" sesi oynatılır.

#### - Seviye Fonkisyonu - Sistem.Seviye[]

iDeal programında bir strateji yazıldığı zaman PERFORMANS veya GETİRİ HESAPLA yöntemletiyle sistemin geçmiş tarihlerdeki başarısı ölüçülürken, her bir al/sat sinyali için işleme giriş fiyatı olarak AKSİ BELİRTİLMEDİĞİ MÜDDETÇE o sinyalin geldiği barın KAPANIŞ FİYATI baz alınır. Bazı sistemlerde KAR AL ve/veya STOP seviyesi girilir ve bu sinyaller geldiğinde

ve sistem stop olduğunda aslında stop sinyali gelen barın kapanış fiyatından değil, stop için girilen seviyeye denk gelen fiyattan stop olunduğu varsayımıyla performans ölçmek gerekir. İster bu nedenle isterse kendi tercihiniz olarak sinyalin olduğu barın kapanışı dışında bir fiyattan (mesela sonraki barın açılılından) işlem olmuş gibi getiri hesabı yapmak mümkündür. Bu ihtiyaç için Sistem.Seviye komutu kullanılır. Seviye komutuna sinyal gelen bardaki işlemin hangi fiyattan gerçekleştiğinin baz alınması kodu yazan tarafından verilir.

Örnek bir kullanım aşağıda yer almaktadır.

```
var C = Sistem.GrafikFiyatSec("Acilis");
var MA1 = Sistem.MA(C, "Exp", 10);
var MA2 = Sistem.MA(C, "Exp", 100);
var SonYon = "";
double Fiyat = 0;
float StopPuan = 0.500F; //500 Puan
for (int i = 1; i<Sistem.BarSayisi; i++)</pre>
{
   if (MA1[i-1] < MA2[i-1] && MA1[i] >= MA2[i] && SonYon != "A") // AL
   {
      Sistem.Yon[i] = "A"; // alış
      SonYon = Sistem.Yon[i];
      Fiyat = C[i];
   }
   else if (MA1[i-1] > MA2[i-1] && MA1[i] <= MA2[i] && SonYon != "S") // SAT
   {
      Sistem.Yon[i] = "S"; // satış
      SonYon = Sistem.Yon[i];
      Fiyat = C[i];
   }
   else if (SonYon == "A" && C[i] < Fiyat - StopPuan) // Long için Stop
   {
      Sistem.Yon[i] = "F"; // flat
      SonYon = Sistem.Yon[i];
      Sistem.Seviye[i] = Fiyat - StopPuan;
   }
   else if (SonYon == "S" && C[i] > Fiyat + StopPuan) // Short stop
   {
      Sistem.Yon[i] = "F"; // flat
      SonYon = Sistem.Yon[i];
      Sistem.Seviye[i] = Fiyat - StopPuan;
   }
}
```

- Sistem Birlestirme - Sistem.Birlestir(Sistem1, Sistem2)

Yazmış olduğunuz birden fazla sistemi birleştirip tek bir sistem olarak kullanmak mümkündür. Bunun için var olan 2 fonksiyondan biri SistemBirlestir diğeri ise SistemBirlestirAyniYon komutudur. Kullanım şekillleri aşağıdaki gibidir;

```
Sistem.SistemBirlestir("XXX", "YYY", "ZZZZ","KKKK");
Sistem.SistemBirlestirAyniYon("XXX", "YYY", "ZZZZ","KKKK");
```

Sistem.Birleştir komutu, yazılmış olan bütün sistemlerin her bir bardaki yönlerine bakar, birleştirilmiş olan YENİ SİSTEM için, aynı barda sistemlerin çoğu hangi yöndeyse o yönde sinyal

# ideal

üretir. Örneğin aşağıdaki görselde birleştirilen 3 sistemde ikisi 30.06.2020 tarihinde AL yönüde, biri SAT yönünde olduğu için, birleştirilmiş sistem aynı tarihli barda, çoğunluğun yönü olan AL yönündedir.

Eğer sistemlerin gönderdiği yön satıları eşit ise, yani örneğin birleştirilen 4 sistemin ikisi AL ikisi SAT ise, bu durumda o barlarda birleştirilmiş sistem FLAT (NAKİT) pozisyonda olur.



SistemBirlestirAyniYon komutu ise; TÜM sistemlerin aynı yönde olduğu barda o yöne giren bir yeni sistem oluşturur. Yukarıdaki üç sistem bu kez bu komutla birleştirilince aşağıdaki (son fotodaki) sistem ortaya çıkmaktadır.;



### - Sistem Getir - Sistem.SistemGetir(SistemAdı, Sembol, Periyot)

Herhangi bir sistem, robot veya formül yazarken, önceden yazmış olduğunuz bir başka sistemden bir veriye ihtiyaç duyabilirsiniz. **SistemGetir** fonksiyonu, başka bir sistemin belirtilen bir sembol ve grafik periyodu için tüm bilgilerini okur ve kullanmaya izin verir.

IDEAL sistem modülünde yazılan her formül bir sistem formülü değildir veya olmayabilir. Hesaplaması tamamen size özel bir indikatör de yazabilirsiniz. Diyelim ki kendi özel indikatörünüzü yazdınız kaydettiniz ve grafiklerde kullanıyorsunuz.

Bir başka formül yazarken bu indikatörünüzün, ABCDE hisse senedinin X dakikalık grafiğine uygulanmasıyla doğan sonuçları kullanmak istiyorsunuz. Sistem Getir fonksiyonu bu noktada ihtiyacınız olan komuttur.

Bu fonksiyon, yazılmış diğer formülden elde edilen HER BİLGİYİ çağırır ve kullanır. Yani çağırdığınız sistemin YÖN listesi (al/sat verdiği barlar ve değerler), ÇİZGİLERİ, o sisteme uygulanmış olan herhangi bir indikatörün sonuçları, o sistemin içinde hesaplanan herhangi bir

liste veya o sistemin getiri/KZ sonuçları....Hepsini tek fonksiyonla okutmuş olursunuz ve istediğiniz anda bir başka kodda kullanabilirsiniz.

Hatta okutulan sistemin PARAMETRELER panelindeki verileri bile okuyabilirsiniz.

#### Sistem.Getir("Sistem Adı", Sembol, Periyot); Şeklinde kullanılır.

(Sistem adı, sembol ve periyot, birer değişken gibi tanımlanırsa, parantezi içine direk değişkenlerin isimleri yazılır. Değişken tanımlanmadan kullanılırsa, bilgilerin üçü de çift tırnak içinde yazılır.

En yaygın kullanım şekillerinden biri de, birden fazla sistem arasında tercih veya kıyas yapmak isteyen bir kullanıcının, o sistemlerin her birinin KARZARAR eğrilerini tek bir grafikte alt alta (veya üst üste) çizdirip durumu görmeleridir.

ÖRNEK 1: TOMA, MACD ve Bollinger isimli sistemlerin GARAN hisse senedinin 60 dakika periyotlu grafiği için hesapladığı KARZARAR listelerini okutup aynı panele çizdirme örneği. (garan grafiğini açıp, tek tek bu sistemleri çalıştırmak yerine, üçünü de okutup, üçünün içindeki GetiriHesaplaKZ listelerini çekmek daha kolay. Hatta ek olarak, bu sistemlerden birinin (Sistemim1 ismiyle çağırdığımız TOMA nın, yön listesini de kullandık. YANİ GRAFİKTE GÖRÜLEN AL/SAT NKTALARI TOMA sisteminden gelenler)

```
var Sistemim1 = Sistem.SistemGetir("TOMA", "IMKBH'GARAN", "G") ;
var Sistemim2 = Sistem.SistemGetir("BolingerSys", "IMKBH'GARAN", "G") ;
var Sistemim3 = Sistem.SistemGetir("MACD", "IMKBH'GARAN", "G") ;
Sistem.GetiriKZ = Sistemim1.GetiriKZ ;
Sistem.GetiriKZ = Sistemim2.GetiriKZ ;
Sistem.GetiriKZ = Sistemim3.GetiriKZ ;
Sistem.Cizgiler[0].Deger = Sistemim1.GetiriKZ; //Sistemim1 (TOMA) Kar Zarar grafiği
Sistem.Cizgiler[1].Deger = Sistemim2.GetiriKZ; //Sistemim2 (Bolinger) Kar Zarar grafiği
Sistem.Cizgiler[2].Deger = Sistemim3.GetiriKZ; //Sistemim2 (Bolinger) Kar Zarar grafiği
```

Sistem.Yon = Sistemim1.Yon;

No	Açıklama	Aktif	Panel	Renk	Kalinlik	Stil
0	TOMA_KZ	$\checkmark$	2		1	1 : Düz
1	Bolinger_KZ	$\checkmark$	2	•	1	1 : Düz
2	MACD_KZ	$\checkmark$	2	•	1	1 : Düz

Grafikteki Görüntü Aşağıdaki gibidir.

## 



ÖRNEK 2: MA\_RSI\_MOMENTUM isimli sistemimizin, 60 dakikalık VIOP grafiğine uygulanması durumunda, en son bardaki yönünü okutup kullanmak... (Bu örnekte, sistemin Son Yönü o sistem içinde parametrelerin sıfırıncı satırına yazdırılıyor. Bu kodda ise, o sistemin ilgili parametre satırı okunup, duruma göre kullanılıyor.

```
var digersistem = Sistem.SistemGetir("MA_RSI_Momentum","VIP'VIP-X030", "60" );
var durum = digersistem.Parametreler[0];
if (durum == "S")
{
    //Diğer Sistem SHORT
    Sistem.Mesaj("Diger Sistem Short");
}
else if (durum == "A")
```

```
{
    //Diğer Sistem LONG
    Sistem.Mesaj("Diger Sistem Long");
}
```

### - Son Fiyat - Sistem. SonFiyat(Sembol)

Bir sembolün o an ki **son fiyatını** okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) son fiyatı okutulmak istenen sembol yazılır.

Not: iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir. Örneğin USDTYR için FX'USDTRY şeklinde yazılır. Bir sembolün PİYASA kodunun ne olduğu, o sembolü sayfanıza yazarken @ işareti yanında gösterilir.)

Örnek kullanım şekli aşağıdaki gibidir.

```
var Sembol = "IMKBH'GARAN";
var X = Sistem.SonFiyat(Sembol);
Sistem.Mesaj(X.ToString()
```

- Son Fiyat - Sistem. SonHacim(Sembol)

Bir sembolün o an ki **en son hacim değerini** okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) son fiyatı okutulmak istenen sembol yazılır. Buradaki hacim değeri o günün başından itibaren gerçekleşen tüm işlemlerin TL hacmidir.

Örnek kullanım şekli aşağıdaki gibidir.

```
var Sembol = "IMKBH'GARAN";
var X = Sistem.SonHacim(Sembol);
Sistem.Mesaj(X.ToString()
```

- Son Lot - Sistem. SonLot(Sembol)

Bir sembolün o an ki **son işlem adedini** okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) son fiyatı okutulmak istenen sembol yazılır. Buradaki dönüş değeri, o günün en başından itibaren gerçekleşejn tüm işlemlerin ADET olarak toplam hacmidir.

Örnek kullanım şekli aşağıdaki gibidir.

```
var Sembol = "IMKBH'GARAN";
var X = Sistem.SonLot(Sembol);
Sistem.Mesaj(X.ToString()
```

- Son Yön Getir - Sistem.SonYonGetir(Sistem, Sembol, Periyot)

Özellikle teknik analiz göstergelerine ve grafik barlarına dayalı stratejilerde, AL/SAT sinyalleri geldiği zaman sıkça kullanılan kavramlardan biri de YÖN kavramıdır. Bir grafikte bir strateji AL sinyaliyle pozisyon açtığında yönümüz artık AL dır ve bir başka sinyal gelene kadar her bir barda son yön değeri A dır. iDealde yön isimli listeler (SİSTEM.YON kısmında da detaylı anlatılacak) A/S/F harfleriyle doldurulurlar.

A = AL Sinyali

S = SAT (AÇIĞA SAT / SHORT POZ AÇ) Sinyali

F = FLAT (Al veya Sat pozisyonundan) Nakite / Flate geç sinyalidir.

Sistemleri okuyarak, herhangi bir hisse senedinde, varantta veya vadeli sözleşmede alım satım yapan robotlar, sistemlerin son yön bilgisini okurlar ve çalılmaya başladıkları anda kendi pozisyonlarını (ayarlanmış işlem adedi kadar alım veya satım yaparak) grafikten gelen bu yöne eşitlerler.

SonYon fonksiyonunun aşağıda gösterilen iki kullanım şekli vardır;

```
Sistem.SonYonGetir(SistemAdi, Sembol, Periyot);
Sistem.SonYonGetirCanli(SistemAdi, Sembol, Periyot);
```

Her iki kullanım şeklinde, sinyal üreten sistemin ismi, hangi sembolün hangi zaman periyotlu grafiğinden sinyal elde edeceiği bilgisi kullanıcı tarafından girilir.

Birinci kullanım şekli, sistemin uygulandığı grafikteki EN SON (CANLI) barda sinyal oluşmasını dikkate almaz, EN SONDAN BİR ÖNCEKİ BAR yönüne bakar. Sinyallerin son/canlı barda sürekli değişebilme ve bu nedenle sürekş al/sat yapabilme ihtimali nedeniyle çoğu yatırımcı koşulların kesinleşmesi, barın kapanması ile işleme girmek ister ve birinci kullanım şeklini tercih eder.



Yukarıdaki grafiğe uygulanan sistemi okuyarak alım satım yapan bir robot Sistem.SonYonGetir komutu kullanıyorsa SON YÖN = S dir ve robot kendi pozisyonunu SHORT pozisyonda olacak şekilde ayarlar. En son barda gelmiş olan AL sinyalini, henüz o bar kapanmadığıo için dikkate almaz.

Eğer bu sistem Sistem.SonYonGetirCanli komutu kullanılarak robota bağlanmış ise, o zaman çalışır çalışmaz AL işlemi yapacaktır çünkü SON YÖN = A dır.

SON Yon Getir komutunun kullanıldığı örnek bir çalışan robot kalıp kodu aşağıda verilmiştir:

```
var LotSize = 1; //işlem adedi
```

```
var SistemAdi = "MA500"; //sistemizin adu
var GrafikSembolu = "IMKBH'GARAN"; //sistemin sinyal ürettiği grafik sembolü
var GrafikPeriyodu = "1"; //grafiğin priyodu
var EmirSembol = "IMKBH'GARAN";
var MySistem = Sistem.SistemGetir(SistemAdi, GrafikSembolu , GrafikPeriyodu ); //sistemin
adı, grafik sembolü, grafiğin periyodu
var SonYon = Sistem.SonYonGetir(SistemAdi, GrafikSembolu , GrafikPeriyodu ); //sistemin
adı, grafik sembolü, grafiğin periyodu
var SonFiyat = Sistem.SonFiyat(EmirSembol);
var Anahtar = Sistem.Name + "," + EmirSembol;
double IslemFiyat = 0;
DateTime IslemTarih;
var Miktar = 0.0;
var Rezerv = "";
var Pozisyon = Sistem.PozisyonKontrolOku(Anahtar, out IslemFiyat, out IslemTarih);
if (Sistem.Saat.CompareTo("10:00:00") <= 0 || Sistem.Saat.CompareTo("17:59:59") >= 0) //
seans yok işlem yapma
{
}
else
{
   if (SonYon == "F" && Pozisyon != 0) // Flata Geç
     Miktar = -Pozisyon;
   else if (SonYon == "A" && Pozisyon != LotSize) // Al
     Miktar = LotSize - Pozisyon;
   else if (SonYon == "S" && Pozisyon != -LotSize) // Sat
     Miktar = -LotSize - Pozisyon;
    // Emir Gönder
    var Islem = ""
    if (Miktar > 0) {Islem = "ALIS"; Rezerv = "ALIS YAPILDI";}
    if (Miktar < 0) {Islem = "SATIS"; Rezerv = "SATI$ YAPILDI";}</pre>
    if (Islem != "")
    {
        Sistem.PozisyonKontrolGuncelle(Anahtar, Miktar + Pozisyon, SonFiyat, Rezerv);
        Sistem.EmirSembol = EmirSembol ;
        Sistem.EmirIslem = Islem;
        Sistem.EmirSuresi = "KIE"
        Sistem.EmirTipi = "Piyasa";
        Sistem.EmirMiktari = Math.Abs(Miktar);
        Sistem.EmirGonder();
    }
}
```

#### - Sorgu Fonksiyonları - Sistem.Sorguxxxxx

iDeal sistem modülünün en çok kullanılan pencerelerinden biri de SORGU penceresidir. Sorgu pencereresi, formül olarak yazdığımız kritere uyan hisseleri tarayıp ekrana döken bir penceredir. Bu pencere, sorgulama işlemi sonrasında ekrana kullanıcı tarafından belirlenmiş olan sütunlar içeen bir tablo getirir. Kaç tane sütun olacağı, hangi sütunda hangi bilginin olacağı, sütunların yazı mı fiyat mı içerdiği, renkleri sola/sağa/ortaya hizalamaları, başlık olarak ne yazacağı ve sayıların virgülden sonra kaç basamaklı gösterileceği gibi tüm seçimler kullanıcı tarafaından belirlenebilir.

Sorgu formülleri yazarken kullanılabilecek fonksiyonlar ve işlevleri aşağıda açıklanmıştır:

```
Sistem.SorguBaslik[0] = 0 nolu sütunun başlığında ne yazsın
Sistem.SorguOndalik[0] = 0 nolu sütundaki sayılar virgülden sonra kaç basamak gösterilsin
Sistem.SorguDeger[0] = 0 nolu sütunda neyin değeri dökülsün
Sistem.SorguSutunTip[0] = 0 nolu sütun YAZI mı FİYAT mı gösterecek
Sistem.SorguSutunHizala[0] = 0 nolu sütundaki veriler nasıl hizalansın (SAG, SOL, ORTA)
```



Sistem.SorguSutunGenislik[0] = 0 nolu sütunun genişliği kaç pixel olsun (Örn: 100)
Sistem.SorguHucreZeminRengi[0] = 0 nolu sütunun zemin rengi ne olsun (Örn: Color.Green)
Sistem.SorguAciklamaGenislik = Açıklama sütununun genişiği kaç pixel olsun (Örn: 120)
Sistem.SorguAciklama = Açıklama sütununda ne yazsın (Örn: "Filtrem")
Sistem.SorguEkle(); Bu satır eklenirse sorgu tarama yapar, aksi halde yapmaz.

Örnek: RSI indikatörüni kendi seçeceğimiz hisseler için ve yine kendi seçeceğimiz grafik periyotları için tarayan, RSI değeri 30 dan küçük, 70 den büyük veya iki seviye arasında olanlara göre renklendirerek tarayan sorgu formülü;

```
// Sütun Tanımları
Sistem.SorguAciklamaGenislik = 1;
Sistem.SorguBaslik[0] = "Kapanış";
Sistem.SorguOndalik[0] = 2;
Sistem.SorguBaslik[1] = "RSI";
Sistem.SorguBaslik[2] = "Is1 Haritas1";
Sistem.SorguSutunTip[2] = "YAZI"; // YAZI
Sistem.SorguBaslik[3] = "Yorum";
Sistem.SorguSutunTip[3] = "YAZI"; // YAZI
Sistem.SorguSutunHizala[3] = "SOL"; // SAG, SOL, ORTA
Sistem.SorguSutunGenislik[3] = 150;
// Sütun Değerleri
var C = Sistem.GrafikFiyatSec("Kapanis");
var RSI = Sistem.RSI(14);
var SonBar = Sistem.BarSayisi - 1;
Sistem.SorguDeger[0] = C[SonBar];
Sistem.SorguDeger[1] = RSI[SonBar];
Sistem.SorguDeger[3] = "RSI = " + RSI[SonBar].ToString("0.00");
if (RSI[SonBar] > 70)
      Sistem.SorguHucreZeminRengi[2] = Color.Green;
else if (RSI[SonBar] < 30)</pre>
    Sistem.SorguHucreZeminRengi[2] = Color.Red;
else if (RSI[SonBar] > 30 && RSI[SonBar] < 50 )
    Sistem.SorguHucreZeminRengi[2] = Color.DarkRed;
else if (RSI[SonBar] > 50 && RSI[SonBar] < 70 )</pre>
    Sistem.SorguHucreZeminRengi[2] = Color.DarkGreen;
Sistem.SorguEkle();
```

Bu sorgu çalıştırılınca aşağıdaki ekran görüntüsü görülür.

Sorgu	(eni	•			Sistem Sorgu					
			XU-100	For	nul Menu	🔲 Güncelle 🛛 900 sanij	/e Hesapla Du			
□ F [	□ 5S □ 10S □ 19	55 🗹 1 🗆	2 🗆 3 🗆 4 🖌	2 5 🗆 8 🗆	10 🗆 15 🗆	20 🗌 30 🗌 60 🔲 120	🗆 240 🗆 S 🗹 G			
No	Sembol	Periyot	Kapanış	RSI	Isı Haritası	Yorum				
1	AKBNK	1	6.28	53.9361		RSI = 53.94	1			
2	AKBNK	5	6.28	46.6433		RSI = 46.64				
3	AKBNK	G	6.28	54.6830		RSI = 54.68				
4	ARCER	1	29.66	48.6081		RSI = 48.61				
5	ARCEK	5	29.66	49.3946		RSI = 49.39				
0	AKCEN	1	29.00	54.1279 45 7093		RSI = 54.15 RSI = 45.71				
8	ASELS ASELS	5	18.07	43,7003		RSI = 43.71 RSI = 43.44				
9	ASELS	Ġ	18.07	55,3292		RSI = 55.33				
10	BIMAS	ĩ	72.75	44.8208		RSI = 44.82				
11	BIMAS	5	72.75	39.4519		RSI = 39.45				
12	BIMAS	G	72.75	63.4627		RSI = 63.46				
13	DOHOL	1	3.07	49.6606		RSI = 49.66				
14	DOHOL	5	3.07	46.0946		RSI = 46.09				
15	DOHOL	G	3.07	63.1387		RSI = 63.14				
16	EKGYO	1	2.13	46.9318		RSI = 46.93				
17	EKGYO	5	2.13	39.7653		RSI = 39.77				
18	EKGYO	G	2.13	56.7056		RSI = 56.71				
19	EREGL	1	13.61	33.7030		RSI = 33.70				
20	EREGL	2	13.61	39.3878 63.3050		RSI = 39.39				
21	CADAN	1	15.01	57 9490		RSI = 05.40 DCT = 57.05				
22	GARAN	5	9,59	47 2835		PSI = 47.28				
24	GARAN	Ğ	9,59	61,2045		RSI = 61.20				
25	GUBRF	ĩ	46.16	38.2091		RSI = 38.21				
26	GUBRF	5	46.16	28.8525		RSI = 28.85				
27	GUBRF	G	46.16	50.5381		RSI = 50.54				
28	HALKB	1	5.38	61.1386		RSI = 61.14				
29	HALKB	5	5.38	50.0534		RSI = 50.05				
30	HALKB	G	5.38	51.7358		RSI = 51.74				
31	ISCTR	1	6.58	54.4175		RSI = 54.42				
32	ISCTR	5	6.58	46.1497		RSI = 46.15				
33	ISCIR	G	6.58	52.2353		RSI = 52.24				
34	KCHOL	1	19.23	39.9955		RSI = 40.00				
35	KCHOL	6	19.23	57.0229		$R_{51} = 57.02$				
30	KO744	1	14 11	50.5099 67 4804		RSI = 67.48				

### - Saözcük Tablosu - Sistem.SozcukTablosunu Oku/Guncelle()

iDeal veri terminali üzerinde robotlar çalıştırıken çoğu zaman bazı bilgileri bir dosyada veya bellekte saklamak ve gerektiğinde kullanmak gerekir. Kod yazılırken bilgi atanan (var x = 5; şeklinde tanımlanan) değişkenler bazen bu ihtiyacı karşılamaz.

Çünkü kod içerisinde tanımlanan değişkenler, kodun her seferinde dafalarca çalışması durumunda hep ilk tanımladığı değerine döner ve ilerleyen satırlarda hesaplamalar sonucu yeni değeri bulunur. Ama bu döngü her seferinde yeniden olur. Oysa istenen şey, bir kez değeri değiştirilen bir verinin, bir daha kullanıcının tanımladığı bir koşul olmadan hiç değişmemesidir. Bu nedenle iDealde Pozisyon Kontrol, Sayı Tablosu, Sözcük Tablosu gibi veri saklama işlevi olan komutlar kullanılır.

Sözcük Tablosu, sözel/text verilerin saklandığı bir bellek değişkenidir. Kod içerisinde sadece istenen koşullar gerçekleştiği zaman değeri değiştirilir. Kod her çalşıştığında yeniden başlangıuçta tanımlanan değeri almaz.

Sözcük Tablosunda saklanan bilgiler RAM de durur ve iDeal kapanırsa bu bilgiler sıfırlanır.

ÖRNEK: Üzerine uygulandığı grafiğin bar değerlerini bir dosyaya yazan, bunu kullanıcıdan onay alarak sadece 1 kez yapan kor örneği.

```
Aktarildimi
                        Sistem.SozcukTablosunuOku(Sistem.Sembol
                                                                        Sistem.Periyot
var
"DosyayVeriAktar");
if (Aktarildimi != "OK")
{
   Sistem.SozcukTablosunuGuncelle(Sistem.Sembol + Sistem.Periyot + "DosyayVeriAktar"
"ОК");
  if
        (MessageBox.Show("Dosyaya
                                     Veri
                                             Aktarılsın
                                                          mı
                                                                (?)",
                                                                         "Veri
                                                                                 Aktarım",
MessageBoxButtons.YesNo) == DialogResult.Yes)
   {
      var Time1 = DateTime.Now; // süre ölçümü için
      var Bars = Sistem.GrafikVerileri;
      var Liste = new List<string>();
      for (var i=0; i<Bars.Count; i++)</pre>
      {
         Liste.Add(Bars[i].Date.ToString("dd.MM.yyyy")
           + ";" + Bars[i].Date.ToString("HH:mm")
           + ";" + Bars[i].Open.ToString()
           + ";" + Bars[i].High.ToString()
           + ";" + Bars[i].Low.ToString()
           + ";" + Bars[i].Close.ToString()
           + ";" + Bars[i].Size.ToString("0"));
      }
      var DosyaAdi = "\\Ideal\\Rapor\\ <file://Ideal/Rapor/>" + Sistem.Sembol + "_
Sistem.Periyot + ".Txt";
      File.WriteAllLines(DosyaAdi, Liste);
      Sistem.Mesaj(Sistem.Sembol + "\r\nPeriyot = " + Sistem.Periyot + "\r\n DOsya Adı = "
+ DosyaAdi + "\r\nSüre = " + (DateTime.Now-Time1).Milliseconds.ToString());
   }
}
```

- Sum - Sistem.Sum()

Bir listenin bütün elemanlarının toplamını veya o listenin her son x elemanının toplamından elde edilen yeni bir liste hesaplatmak için Sistem.Sum fonksiyonları kullanılır. İki kullanım şekli vardır;

**Sistem.Sum(Liste)** : Girdi olarak verilen bir listenin bütün elemanlarının toplam sayısal değerini bulur. Örnek bar kapanışlarının listesi C ise; Sistem.Sum( C ) dersek, o grafikteki barların hepsinin close değerlerinin toplamı bir sayı olarak bize verilir.

**Sistem.Sum(Liste, 10)** : Girdi olarak verilen bir listenin hep son elemanının toplam değerinden oluşan yeni bir liste veriri. Örnek bar kapanışlarının listesi C ise; Sistem.Sum(C, 10) dersek, o grafikteki barların her birinin son bardaki değerleri toplar ve yeni liste/çizgi/indikatör elde eder.

ÖRNEK: Fiyat önceki bara göre daha küşük kapanış yaparsa değeri "-1" olan bir XX listesi oluşturup, bunun 5 periyotlu SUM fonksiyonunu hesaplatalım;

Bu sonuç indikatörü 0 ile -5 arasında değerler alan bir indikatör olacak. Bu durumda değerin -5 olduğu bar bize fiyatın 5 bardır düştüğü bilgisini vereektir;

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var xx = Sistem.Liste(0);
for (int i=3; i < Sistem.BarSayisi; i++)
{
    xx[i] = C[i] <= C[i-1] ? -1 : 0;
}
var sum = Sistem.Sum(xx,5);
```

### - Süper Trend - Sistem.SuperTrend()

iDeal indikatör kütüphanesinde yer alan Süper Trend isimli indikatörü çağırır. 3 adet paramatre alır ve varsayılan değer olarak 3/10/14 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.SuperTrend(Factor, HHVPeriyot, ATRPeriyot);
Sistem.SuperTrend(Veriler, Factor, HHVPeriyot, ATRPeriyot);
```

### - Standart Deviation (Standart Sapma) - Sistem.StDev()

iDeal indikatör kütüphanesinde yer alan Standart Deviation (standart sapma) isimli indikatörü çağırır. 2 adet paramatre alır ve varsayılan değer olarak 5/1 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

Sistem.StDev(5, 1)
Sistem.StDev(Verileri, 5, 1)

Ayrıca 2 amaç için daha standart sapma fonksiyon kullanımı sunulmuştur. Bunlardan biri bir listenin tüm elmanlarının stnadart sapması, diğeri ise, listenin son x elemanının standart sapmasından hesaplanan yeni bir liste sunan fonksiyondur. Bu iki fonksiyonun kullanım şekilleri aşağıda belirtildiği gibidir.

```
Sistem.StDev(Liste, Deger) (örneğin close listesinin son 5 elemanının standart sapmaları)
Sistem.StDevDeger(Liste) (Örneğin tüm close'ların standart sapmasını hesapla)
```

### - Standart Error - Sistem.StEr()

iDeal indikatör kütüphanesinde yer alan Standart Error isimli indikatörü çağırır. 1 adet paramatre alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.StEr(14)
Sistem.StEr(Veriler , 14)
```

### - Standart Error Bands - Sistem.StErDown/Mid/Up()

iDeal indikatör kütüphanesinden yer alan Standart Error Bands isimli indikatörü çağırır. 4 adet paramatre alır ve varsayılan değer olarak 14/2/Simple/1 kullanılır. Bu indikatör Üst/Alt ve Orta çizgisi olan bir band oscilatördür. Her bir çizgisi için ayrı ayrı ideal sistem fonksiyonu

vardır. İsteyen sadece bandın bir çizgisi çağırıp/çizdirip/kullanabilir. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.StErDown(14, 2,"Simple", 1)
Sistem.StErDown(Veriler , 14, 2,"Simple", 1)
Sistem.StErMid(14, 2,"Simple", 1)
Sistem.StErMid(Veriler , 14, 2,"Simple", 1)
Sistem.StErUp(14, 2,"Simple", 1)
Sistem.StErUp(Veriler , 14, 2,"Simple", 1)
```

```
- Stochastic Fast - Sistem.StochasticFast(5,3)
```

iDeal indikatör kütüphanesinde yer alan Stochastic Fast isimli indikatörü çağırır. 2 adet paramatre alır ve varsayılan değer olarak 5/3 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.StochasticFast(5, 3);
Sistem.StochasticFast(Veriler , 5, 3);
```

- Stochastic Momentum Index - Sistem.StochasticMomIndex(5,3,3)

iDeal indikatör kütüphanesinde yer alan Stochastic Momentum Index isimli indikatörü çağırır. 3 adet paramatre alır ve varsayılan değer olarak 5/3/3 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.StochasticMomIndex(5,3,3)
Sistem.StochasticMomIndex(Liste,5,3,3)
Sistem.StochasticMomIndex(Veriler, 5,3,3)
```

- Stochastic Oscillator - Sistem.StochasticOsc(5,3)

iDeal indikatör kütüphanesinde yer alan Stochastic Oscillator isimli indikatörü çağırır. 2 adet paramatre alır ve varsayılan değer olarak 5/3 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.StochasticOsc(5,3)
Sistem.StochasticOsc(Veriler, 5,3)
```

- Stochastic RSI - Sistem.StochasticRSI(14)

iDeal indikatör kütüphanesinde yer alan Stochastic RSI isimli indikatörü çağırır. 1 adet paramatre alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.StochasticRSI(14)
Sistem.StochasticRSI(Liste,14)
Sistem.StochasticRSI(Veriler, 14)
```

- Stochastic Slow - Sistem.StochasticSlow(5,3)

iDeal indikatör kütüphanesinde yer alan Stochastic Slow isimli indikatörü çağırır. 2 adet paramatre alır ve varsayılan değer olarak 5/3 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;



Sistem.StochasticSlow(5,3); Sistem.StochasticSlow(Veriler, 5,3);

#### <u>Stop veya KarAl Puan/Yüzde - Sistem.StopVeyaKarAl(stop,kar)</u>

Yazdığınız sistemin belli kriterler geçekleşirse pozisyondan çıkıp nakite geçmesini ve bir sonraki sinyale kadar nakitte beklemesini sağlar. Burada nakite geçme (FLAT Olma) kriteri ya belli bir puan veya yüzde miktarda kar elde edilmesi ya da işleme girdikten sonra gördüğü en iyi/karlı noktadan belli bir puan veya yüzde geri dönmesi olabilir

Bu fonksiyon iki parametre ile birlikte kullanılır. İlk parametre Stop, ikinci parametre ise KAR AL seviyesi için girilir:

**STOP SEVİYESİ:** Sistem işlem yaptıktan sonra, gördüğü en iyi fiyat seviyesinden, buraya girilen puan/yüzde kadar geri dönerse, sistem stop yapar ve FLAT pozisyona (nakite) geçer.

İşlem yapar yapmaz terse giden bir sistem, bu parametre kadar ters yöne giderse ve halen ana strateji sinyal üretmemişse buraya girilen seviye kadar zararda STOP yapar

**KAR AL SEVİYESİ:** Sistem işlem yaptıktan sonra buraya girilen parametredeki puan/yüzde kadar kar elde edilirse, sistem karı alır ve FLAT pozisyona geçer.

Bu parametrelerden herhangi birisi kullanılmak istenmiyorsa "0" girilir. İki parametre aynı anda kullanılabilir. Önce hangi şart sağlanırsa o şartın oluştuğu yerde STOP olur.

ÖRNEK: 50 VE 200'lik iki Exponatial ortalamanın kesişimlerine göre sinyal üreten sistemimize, 4500 puan kar elde edince veya işleme girdikten sonra görülen en iyi fiyat seviyesinden 1500 puan terse dönüş olursa pozisyonu kapat eklediğimiz örnek.

```
var Yontem = "Exp";
var Periyot1 = 50;
var Periyot2 = 200;
// kapanış fiyatlarını oku
var C = Sistem.GrafikFiyatSec("Kapanis");
// hareketli ortalamaları hesapla
var MA1 = Sistem.MA(C, Yontem, Periyot1);
var MA2 = Sistem.MA(C, Yontem, Periyot2);
```

```
Sistem.KesismeTara(MA1, MA2);
Sistem.StopVeyaKarFlatPuan(1.500,4.500);
```



### - Swing Index - Sistem.SwingIndex(3)

iDeal indikatör kütüphanesinde yer alan Swing Index isimli indikatörü çağırır. 1 adet paramatre alır ve varsayılan değer olarak 3 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.SwingIndex(3);
Sistem.SwingIndex(Veriler, 3);
```

- Taban Fiyat - Sistem.Taban(Sembol)

Bir sembolün o günkü TABAN fiyatını okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) taban fiyatı okutulmak istenen sembol yazılır.

Örnek kullanım şekli aşağıdaki gibidir.

```
var Sembol = "IMKBH'GARAN";
var TabanFiyat = Sistem.Taban(Sembol);
Sistem.Mesaj(TabanFiyat.ToString());
```

### - Tavan Fiyat - Sistem.Tavan(Sembol)

Bir sembolün o günkü TAVAN fiyatını okumak için kullanılır. Parametre olarak fonksiyona (parantez içine) taban fiyatı okutulmak istenen sembol yazılır.

Örnek kullanım şekli aşağıdaki gibidir.

```
var Sembol = "IMKBH'GARAN";
var TavanFiyat = Sistem.Tavan(Sembol);
Sistem.Mesaj(TavanFiyat.ToString());
```

- Tablo Kullanımları - Sistem.Tablo...(Sembol)

Ekrana satırları, sütunları kullanıcı tarafından belirlenebilen tablolar açmak isteyen ve ister programdan okutulan isterse hesaplatılan verilerle de bu tabloların için doldurmak isteyen kullanıcılarımız için 3 adet tablo fonksiyonu sunulmuştur. Bu fonksiyonların yazım şekilleri ve işlevleri aşağıda belirtilmiştir;

```
Sistem.Tablo(TabloAdi, Sol, Sag, Genislik, Yukseklik, SutunSayisi, SatirSayisi,
SutunGenislik, SutunHizala, SutunBaslik)
Sistem.TabloTemizle(TabloAdi)
Sistem.TabloYazdir(TabloAdi, Sutun, Satir, Metin, ZeminRenk, YaziRenk)
```

### Sistem.Tablo Fonksiyonu:

Ekrana bir tablo açılmasını sağlayan komuttur. Ekrana bir tablo açtırmadan önce 12 adet parametreyi sisteme tanıtmak gerekir. Bu paramatreler fonksiyonun yukarıda belirtilen yazıum şeklindeki sırayla girilir ve hangi parametrenin ne işe yaradığı bilgisi aşağıda açıklanmıştır;

TabloAdi = Ekrana açılacak tablonun ismi (hem ekrana açılan tablonun başlığında bu isim yazılı gözükür hem de formüllerde tablo açtırılırken, temizlenirken, yazdırılırken bu isim kullanılır) Sol = Tablo, monitörde ekranın en solunda kaç pixe uzakta olsun bilgisidir (örnek olarak 100 gibi bir sayı girilebilir.)

Sag = Tablo, monitörde ekranın en üstünden kaç pixel aşağıda/uzakta olsun bilgisidir (örnek olarak 100 gibi bir sayı girilebilir.)

**Genislik** = Açılacak tablonun genişliği kaç pixel olacak bilgisidir. (Örnek olarak 400 gibi bir sayı girilebilir)

**Yukseklik** = Açılacak tablonun yüksekliği kaç pixel olacak bilgisidir (Örnek olarak 600 gibi bir sayı girilebilir.)

SutunSayisi = Tabloda kaç adet sütun olacağı bilgisidir.

SatirSayisi = Tabloda kaç adet sütun olacağı bilgisidir.

**SutunGenislik** = HER BİR SÜTUN için ayrı ayrı olarak sütun genişliği kaç pixel olacak bilgisi bu parametre ile belirlenir. Kaç adet sütun tanımlanmışsa o sayı kadar elemanı olan bşr liste şeklinde tanıtılmalıdır.

Örnek: 4 sütunlu bir tablo açacağız. O zaman bu 4 sütunun her birinin pixel cinsinden genişliklerini aşağıdaki gibi sisteme tanımlarız;

var GenisLikler = new int[4]{90,70,120,150};

Yukarıdaki satır şu demektir; her biri int (integer = tamsayı) tipinde olan 4 elemanlı bir listemiz var ve listemizin elemanları (aslında sütunlarımızın genişlik değerleri) sırasıyla 90, 70, 120 ve 150 pixeldir. Bu tanımı yaptıktan sonra Sistem.Tablo fonksiyonu içindeki SütunGenislik parametresi olan değer bir sayı değil bu sayılar listesini tanımladığımız Genislikler değişkeninin ismi yazılır.

**SutunHizala** = HER BİR SÜTUN için ayrı ayrı olarak, sütunlarda yer alacak bilgilerin sola yaslı, sağa yaslı veya ortalı olarak hizalama tipi bilgisi bu parametre ile belirlenir. Kaç adet sütun tanımlanmışsa o sayıda kadar elemanı olan bir liste şeklinde tanıtılmalıdır.

Örnek: 4 sütunlu bir tablo açacağız. O zaman bu 4 sütunun her birinin hizalama tipini aşağıdaki gibi sisteme tanımlarız;

var Hizalamalar = new int[4]{0,1,1,2};

- 0 = Sola Yaslı
- 1 = Ortala
- 2 = Sola Yaslı

Yukarıdaki satır şu demektir; her biri int (integer = tamsayı) tipinde olan 4 elemanlı bir listemiz var ve listemizin elemanları (aslında sütunlarımızın hizalama seçimleri) sırasıyla 0, 1, 1 ve 2 dir. Bu tanımı yaptıktan sonra Sistem.Tablo fonksiyonu içindeki SütunHizala parametresi olan değer bir sayı değil bu sayılar listesini tanımladığımız Hizalamalar değişkeninin ismi yazılır. **SutunBaslik** = HER BİR SÜTUN için ayrı ayrı olarak, sütunların başlıklarının ne olacağı bilgisi bu parametre ile belirlenir. Kaç adet sütun tanımlanmışsa o sayıda kadar elemanı olan bir liste şeklinde tanıtılmalıdır.

Örnek: 4 sütunlu bir tablo açacağız. O zaman bu 4 sütunun her birinin başlık bilgisi aşağıdaki gibi sisteme tanımlarız;

var Basliklar = new string[4]{"Sembol","Son Fiyat","Önceki Kapanış","Fark %"};

Örnek: Aşağıdaki kod satırları kaydedilip bir kez çalıştırlırsa (FORMÜL TEST butona basılırsa) ekrana aşağıdaki gibi boş bir tablo açılacaktır.

```
string TabloAd = "YUKSELENLER ";
var SutunGenislik = new int[4]{90,70,120,150};
var SutunHizala = new int[4]{0,1,1,1}; //0= sola yasla 1=ortala 2=sağa yasla
var SutunBaslik = new string[4]{"Sembol","Son Fiyat","Önceki Kapanış","Fark %"};
Sistem.Tablo(TabloAd, 100, 100, 500, 500, 4, 500, SutunGenislik, SutunHizala, SutunBaslik);
```

YUKSELENLER				×
Sembol	Son Fiyat	Önceki Kapanış	Fark %	
				E
				]

### Sistem.TabloTemizle Fonksiyonu:

Ekrana açtırılmış bir tablonun, aralıklarla veya bir koşula bağlı olarak içindeki verilerin temizlenmesi amacıyla kullanılır. Kod akışında birtakım verileri elde edip tabloya yazdırıp, örneğin Saat = 11:00:00 ise tabloyu temizle gibi bir kod satırı yazılabilir.

### Sistem.TabloYazdir Fonksiyonu:

Ekrana açtırılacak olan tablonun hücrelerine verilerin yazdırılması için kullanılan komuttur. Aşağıdaki şekilde kullanılır ve satırlar/sütunlara bilgiler yazdırılır.

Sistem.TabloYazdir(TabloAdi, Sutun, Satir, Metin, ZeminRenk, YaziRenk);

Tablonun birden fazla satırı ve/veya sütunu olduğu için, tek tek her bir satır ve sütun (aslında hücre) için çok sayıda satır kod yazarak da veriler doldurulabilir. Ama yazılım dili kullanmanın bir avantajı, sürekli tekrar eden ve belli kuralları işlemleri DÖNGÜ içinde yapabilmektedir.

Örnek1: GARAN, ISCTR, AKBNK ve YKBNK senetlerinin Son İşlem, Taban ve Tavan fiyatlarını ekranda açtırılacak bir tabloya, her bir hücreyi tek tek yazdırmak (döngü kullanmadan)

```
string TabloAd = "TABLOM ";
var SutunGenislik = new int[4]{90,70,100,150};
var SutunHizala = new int[4]{0,1,1,1}; //0= sola yasla 1=ortala 2=sağa yasla
var SutunBaslik = new string[4]{"Sembol","Son Fiyat","Taban","Tavan"};
Sistem.Tablo(TabloAd, 100, 100, 450, 400, 4, 300, SutunGenislik, SutunHizala, SutunBaslik);
var GARANVeri = Sistem.YuzeyselVeriOku("IMKBH'GARAN");
var AKBNKVeri = Sistem.YuzeyselVeriOku("IMKBH'AKBNK");
var ISCTRVeri = Sistem.YuzeyselVeriOku("IMKBH'ISCTR");
var YKBNKVeri = Sistem.YuzeyselVeriOku("IMKBH'YKBNK");
Sistem.TabloYazdir(TabloAd, 0, 0, "GARAN", Color.White, Color.Blue);
Sistem.TabloYazdir(TabloAd, 1, 0, GARANVeri.LastPrice.ToString(), Color.White, Color.Blue);
```

Sistem.TabloYazdir(TabloAd, 2, 0, GARANVeri.LimitDown.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 3, 0, GARANVeri.LimitUp.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 1, "ISCTR", Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 1, 1, ISCTRVeri.LastPrice.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 2, 1, ISCTRVeri.LimitDown.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 3, 1, ISCTRVeri.LimitUp.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 0, 2, "AKBNK", Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 1, 2, AKBNKVeri.LastPrice.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 2, 2, AKBNKVeri.LimitDown.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 3, 2, AKBNKVeri.LimitUp.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 3, 2, AKBNKVeri.LimitUp.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 3, 2, AKBNKVeri.LimitUp.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 1, 3, YKBNK", Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 1, 3, YKBNKVeri.LastPrice.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 1, 3, YKBNKVeri.LimitDown.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 3, 3, YKBNKVeri.LimitDown.ToString(), Color.White, Color.Blue); Sistem.TabloYazdir(TabloAd, 3, 3, YKBNKVeri.LimitDown.ToString(), Color.White, Color.Blue);

TABLOM				х
Sembol	Son Fiyat	Taban	Tavan	
GARAN	10.03	9.04	11.04	1
ISCTR	6.73	6.04	7.38	
AKBNK	6.5	5.82	7.1	
YKBNK	3.06	2.75	3.35	
				- 1

Örnek2: Adı XXXX olan stratetejimizi YILDIZ pazardaki tüm hisseler için tara, her bir hisse için performans testi yap ve her bir hissedeki performans sonuçlarını ekrana bir tablo olarak getir formülü. (Not: XXX isimli sistemin en son satırına Sistem.GetiriHesapla("01/01/1990", 0.0); satırı eklenmiş olmalı ki, getiri hesaplanabilsin)

Kodu çalıştırınca ekrana aşağıdaki tablo açılır.

```
var Periyot = "G";
var Sistem_ADI = "XXX";
var Semboller = new List<string>();
var Liste = Sistem.YuzeyselListeGetir("IMKBH'");
for (int i = 0; i < Liste.Count; i++)</pre>
{
   if (Liste[i].Grup == "Y" && Liste[i].Seri == "E")
      Semboller.Add(Liste[i].Symbol);
}
//ekrana açılacak tablonun satır, sütun, genişlik, hizalama vs bilgilerini gir
string TabloAd = Sistem_ADI + " Sisteminin" + " " + "GETIRI TABLOSU ";
var SutunGenislik = new int[4]{120,110,90,90};
var SutunHizala = new int[4]{0,1,2,2};
var SutunBaslik = new string[4]{"Sembol","Karlı İşlem Oranı","Net Getiri","İşlem Sayısı"};
Sistem.Tablo(TabloAd, 100, 100, 460, 510, 4, 200, SutunGenislik, SutunHizala, SutunBaslik);
for (var No=0; No < Semboller.Count; No++)</pre>
{
   var Sembol = Semboller[No];
   var s = Sistem.SistemGetir(Sistem_ADI,Sembol, Periyot);
   Sistem.GetiriKZ = s.GetiriKZ ;
   Sistem.GetiriToplamIslem = s.GetiriToplamIslem;
   Sistem.GetiriKarIslemOran = s.GetiriKarIslemOran;
   var getiri = Sistem.GetiriKZ[Sistem.GetiriKZ.Count-1];
   var islemsayisi = Sistem.GetiriToplamIslem;
```

```
var KarliIslemOran = Sistem.GetiriKarIslemOran;
//tabloyu doldur
Sistem.TabloYazdir(TabloAd, 0, No, Sembol, Color.White, Color.Blue);
Sistem.TabloYazdir(TabloAd, 1, No, KarliIslemOran.ToString("0.00"), Color.White, Color.Blue);
var Renk = getiri < 0 ? Color.Red : Color.Blue; //NEGATİF OLANLARI KIRMIZI YAP
Sistem.TabloYazdir(TabloAd, 2, No, getiri.ToString("0.000"), Color.White, Renk);
Sistem.TabloYazdir(TabloAd, 3, No, islemsayisi.ToString(), Color.White, Renk);
}</pre>
```

Sembol	Karlı İşlem Oranı	Net Getiri	İşlem Sayısı	
IMKBH'AEFES	40.00	-9.314	165	
IMKBH'AGHOL	38.82	8.597	170	1
IMKBH'AKBNK	44.44	-2.966	171	L
IMKBH'AKCNS	47.86	10.796	140	
IMKBH'AKFGY	43.48	4.770	138	
IMKBH'AKGRT	38.55	-0.014	166	
IMKBH'AKSA	44.03	1.883	159	
IMKBH'AKSEN	36.31	-1.790	168	
IMKBH'AKSGY	50.00	3.225	116	
IMKBH'ALARK	50.00	5.790	146	
IMKBH'ALBRK	42.77	0.949	166	
IMKBH'ALCTL	46.63	12.620	163	
IMKBH'ALGYO	49.04	21.217	157	
IMKBH'ALKIM	40.48	-7.209	168	
IMKBH'ANHYT	48.63	5.819	146	
IMKBH'ANSGR	50.00	1.123	166	
IMKBH'ARCLK	48.32	14.739	149	
IMKBH'ARDYZ	33.33	-11.970	15	
IMKBH'ASELS	40.37	3.075	161	
IMKBH'AVISA	51.22	9.150	82	
IMKBH'AYGAZ	45.73	10.194	164	
IMKBH'BAGFS	43.40	8.930	159	
IMKBH'BIMAS	38.12	-109.932	181	
IMKBH'BIZIM	43.79	1.378	153	
IMKBH'BRISA	43.83	2.034	162	
IMKBH'BRSAN	48.03	18.790	152	

### - Tarih - Sistem.Tarih()

iDeal çalışan bilgisayardan okutulan tarih bilgisini veren sistem fonksiyonudur. YYYY.AA.GG şeklinde string olarak dönüş verir.

Örnek kullanım: ekrana o anki tatihi mesaj olarak getiren kod:

```
var Tarih = Sistem.Tarih;
Sistem.Mesaj(Tarih);
```

ÖRNEK: İndikatörümüz istediğimiz bir tarihten sonra hesaplanıp çizilsin.

```
var BASLAMATARIH = "20190103"; // YYYYAAGG seklinde
var V = Sistem.GrafikVerileri;
var C = Sistem.GrafikFiyatSec("Kapanis");
var 0 = Sistem.GrafikFiyatSec("Acilis");
```



```
var IND = Sistem.Liste(0);
var Baslangicbar = 0;
for (int i=1; i < V.Count; i++)
{
    if (V[i].Date.ToString("yyyyMMdd") == BASLAMATARIH )
    Baslangicbar = i;
}
for (int i=Baslangicbar ; i < V.Count; i++)
    IND[i]= IND[i-1] + (0[i] - C[i-1]);
Sistem.Cizgiler[0].Deger = IND;
```

- Saat Tarih Aralığı - Sistem. Tarih Araligi ("2019.01.20", "2020.12.31")

Belli tarihler arasında bir iş yaptırmak veya yaptırmamak için Tarih Aralığı fonksiyonu kullanuılabilir.

ÖRNEK: Yazılmış bir sistem/robot kodu şifrelenerek bir başkasıyla paylaşılacak ve 2020 yılı 31 Aralık tarihinden sonra çalışmasın isteniyor. Bu fonksiyon ile söz konusu işlem aşağıdaki şekilde sağlanır.

```
if (Sistem.TarihAraligi("2020.01.01", "2020.12.31"))
{
    // KODUN TAMAMI
}
else
    Sistem.Mesaj("Bu Sistemi Kullanmaya Yetkiniz Yoktur. Lütfen iletişime geçiniz.");
```

#### <u>TEMA – Sistem.TEMA(5)</u>

TEMA indikatörünü çağırır. 1 adet parametre (periyot) alır. Aşağıdaki gibi 3 kullanım şekli vardır;

```
Sistem.TEMA(21);
Sistem.TEMA(Liste,21);
Sistem.TEMA(Veriler,21);
var TEMA = Sistem.TEMA(21);
Sistem.Cizgiler[0].Deger = TEMA;
```

- <u>Tillson T3 - Sistem.TillsonT3(9, 0.618)</u>

iDeal indikatör kütüphanesinde yer alan Tillson T3 isimli indikatörü çağırır. 2 adet paramatre alır ve varsayılan değer olarak 9/0.618 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

Sistem.TillsonT3(9, 0.618);
Sistem.TillsonT3(Listre, 9, 0.618);

- TSF / Time Series Forecast - Sistem.TimeSeriesForecast(14)

```
SEZAİ KILIÇ
```

iDeal indikatör kütüphanesinde yer alan Time Series Forecast isimli indikatörü çağırır. 1 adet paramatre alır ve varsayılan değer olarak 14 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

Sistem.TimeSeriesForecast(14);
Sistem.TimeSeriesForecast(Veriler, 14);

### - TKE - Sistem.TKE()

iDeal indikatör kütüphanesinde yer alan TKE isimli indikatörü çağırır. Hiç paramatre almaz. Aşağıdaki gibi yazım şekilleri vardır;

Sistem.TKE();
Sistem.TKE(Veriler);

TKE içerisinde 7 adet indikatör barından bir eğridir. İç formülü de aşağıda paylaşılmıştır. Dileyen kullanıcılar TKE isimli fonksiyon yerine aşağıdaki açık formülü kullanabilir ve isterse iç formüldeki indikatör periyotlarını değiştirerek kullanabilir.

```
var STOFK = Sistem.StochasticOsc(14, 6);
var RSI = Sistem.RSI(14);
var CCI = Sistem.CommodityChannelIndex(14);
var MFI = Sistem.MoneyFlowIndex(14);
var WR = Sistem.WilliamsR(14);
var MOM = Sistem.Momentum(14);
var ULT = Sistem.UltimateOsc(7, 14, 28);
var TKE = Sistem.Liste(0);
for (int i = 0; i < Sistem.BarSayisi; i++)
TKE[i] = (STOFK[i] + RSI[i] + CCI[i] + MFI[i] + WR[i] + MOM[i] + ULT[i]) / 7;
Sistem.Cizgiler[0].Deger = TKE
```

```
- TOMA (Trailing Oscilator Of Moving Average) - Sistem.TOMA(3,2)
```

iDeal indikatör kütüphanesinde yer alan ve aslında bir hareketli ortalamayı, kullanıcı tarafından girilen bir yüzde oranında bir nevi izleyen stop ile takip eden yeni bir çizgiden ibaret olan TOMA isimli indikatörü çağırır. 2 veya 3 parametre girilen 2 ayrı stille çağrılabilir ve bu nedenle toplam 6 şekilde (Aşağıdaki gibi) yazılabilir.

```
Sistem.TOMA(3, 2);
Sistem.TOMA(3, 2 , MAYontem);
Sistem.TOMA(Liste, 3, 2);
Sistem.TOMA(Liste, 3, 2, MAyontem);
Sistem.TOMA(Veriler, 3, 2);
Sistem.TOMA(Veriler, 3, 2, MAyontem);
```

TOMA indikarörü iç formülünde bir MA (Moving Average) vardır ve TOMA için girilmiş olan birinci parametre olan sayı kaç ise (varsayılan 3) o periyotluk bir EMA'dır. EMA = Exponantial Movin Average demektir. Yani kullanılan Hareketli Ortalamanın TÖNTEM bilgisi "Exp" dir. iDeal kullanıcıları TOMA'nın iç formülünde yer alan MA için Simple veya başka bir yöntem kullanabilirler. Bunun için yukarıdaki yazım şekillerinde MAYontem ifadesi olan satırları kullanırız. Örnek: Sistem.TOMA(3,2, "Simple")

ÖRNEK: TOMA kullanılatak al/sat sinyalleri üreten örnek bir sistem;

```
var TomaPeriyot = 3; //EMA TOMA PERİYODU
var Yuzde = 1.8; //TOMA YÜZDESİ
var C = Sistem.GrafikFiyatSec("Kapanis");
var TOMA = Sistem.TOMA(C,TomaPeriyot, Yuzde);
var EMA = Sistem.MA(C, "Exp",TomaPeriyot);
var SonYon = "";
for (int i= 1; i < Sistem.BarSayisi; i++)</pre>
{
   if (EMA[i] > TOMA[i] && SonYon != "A")
   {
       SonYon = "A";
       Sistem.Yon[i] = "A";
   }
   if (EMA[i] < TOMA[i] && SonYon != "F")</pre>
   {
       SonYon = "F";
       Sistem.Yon[i] = "F";
   }
}
Sistem.Cizgiler[0].Deger = EMA;
Sistem.Cizgiler[1].Deger = TOMA;
```

```
- TOMA PUAN - Sistem.TOMAPUAN(3,2)
```

iDeal indikatör kütüphanesinde yer alan TOMA isimli indikatör, kendisine parametre olarak verilen bir YÜZDE ile MA'yı izleyen bir indikatYÜZDE ile değil PUAN (viop veya endeks tarzı enstrümanlar için genelde) olarak takip eden hali de sunulmuştur. Aşağıdaki gibi yazım şekilleri vardır. Klasik TOMA da 3/2 parametreleri 3'lük EMA ile YÜZDE2 izleyen diye yorumlarken, TOMAPUAN kullanımında 3/10 dersek 3'lük EMA ile 10 puan (mesela 1430 puan olan bir endeks veya vadelide kullanınca 10 puan ile izlemek) takibi ile kullanılmış olur.

Sistem.TOMAPUAN(3, 2)(3, 2); Sistem.TOMAPUAN(Liste, 3, 2); Sistem.TOMAPUAN(Veriler, 3, 2);

NOT: Sistem.TOMAS isimli bir komut daha vardır TOMA kullanım şekillerinde. Bu kullanım EMA yerine SMA kullanarak TOMA hesaplamak içindir.

- Trend Çiz - Sistem.TrendCiz()

Grafikler üzerinde kod yazarak Trendler çizbilmek için kullanılan iDeal Sistem Fonksiyonudur. Bir trend çizgisi, tanımlı iki noktadan geçen bir doğru parçasıdır. Dolayısıyla 2 noktayı fonksiyona bilgi olarak girmek gerekir. Söz konusu veri seti grafikler/barlar olunca, sisteme verilmesi gereken noktalardan her birinin hem yatay (zaman) hem dikey (fiyat) eksenindeki konumu verilmek durumdadır. Fonksiyonun yazım/kullanım şekli aşağıdaki gibidir.

```
Sistem.TrendCiz(Tarih1, Deger1, Tarih2, Deger2);
```

ÖRNEK1: Tarih ve Fiyat bilgilerini elle girerek bir trendi çizdirmek


DateTime Tarih1, Tarih2; double Fiyat1, Fiyat2; Tarih1 = new DateTime(2020, 3, 17, 0, 0, 0); // y1l, ay , gün, saat, dakika, saniye Fiyat1 = 1.70; Tarih2 = new DateTime(2020, 10, 30, 0, 0, 0); // y1l , ay , gün , saat , dakika, saniye Fiyat2 = 3.10; var Trend = Sistem.TrendCiz(Tarih1, Fiyat1, Tarih2, Fiyat2); Sistem.Cizgiler[0].Deger = Trend;

Yukarıdaki formül KRDMD günlük grafiğine uygulandığında aşağıdaki trend çizilmiş olacaktır.



ÖRNEK2: Birinci ve İkinci noktanın tarihleri kendi formüllerinizde bazı koşulların sağlandığı tarihler olarak buldurulabilir veya o bulunan noktaların BAR NUMARASI tespit edilebilirç Böyle bir durumda söz konusu koşullara göre dinamik olarak kendi çizilen trendler elde edebilirsiniz. Bu yöntemde bar numarası kullanarak trend çizdirme örneği aşağıdaki gibidir;

```
var V = Sistem.GrafikVerileri;
var baslamabari = 1289; // 1289 nolu bar birinci nokta
var bitisbari = 1400; //1400 nolu bar ikinci nokta
var Trend = Sistem.TrendCiz(V[baslamabari].Date, V[baslamabari].Low, V[bitisbari].Date,
V[bitisbari].Low);
Sistem.Cizgiler[0].Deger = Trend;
```



#### - Trend Paralel Çiz - Sistem.ParalelTrendCiz()

Sistem.Cizgiler[1].Deger = Paralel;

Grafikler üzerinde çizdirilmiş bir trende paralel çizdirmeyi sağlayan sistem fonksiyonudur. Kullanım şekli aşağıdaki gibidir;

Sistem.TrendParalelCiz(Trend, Tarih1, Tarih2)

#### ÖRNEK:

```
DateTime Tarih1, Tarih2;
double Fiyat1, Fiyat2;
Tarih1 = new DateTime(2020, 3, 17, 0, 0, 0); // y1l, ay , gün, saat, dakika, saniye
Fiyat1 = 1.70;
Tarih2 = new DateTime(2020, 10, 30, 0, 0, 0); // y1l , ay , gün , saat , dakika, saniye
Fiyat2 = 3.10;
var Trend = Sistem.TrendCiz(Tarih1, Fiyat1, Tarih2, Fiyat2);
Sistem.Cizgiler[0].Deger = Trend;
var Paralel = Sistem.TrendParalelCiz(Trend, Tarih1, Tarih2);
```

🛪 🗰 KRDMD 🖡 ORJ Gün 🕂 I T TrendCiz 🔶 Oto ideAlgo 5 30 240 💪 Y									
🔚 Skala Periyot	+ → ↔ → ↓ ↓ ↓ ↓ = ⊕	) 🗇 😈							
Mar Nis May Haz Tem Ağu Eyl	Eki Kas Ara 2020 Şub Mar N	is May Haz Tem Ağu Eyl Eki Kas							
Tarih         10.11.2020           Açılış         3.43           Yüksek.         3.75           Düşük         3.42           Kapanış         3.73           Hacim         1.437.385.000           Lot         396.468,300           Oct         2.52		4.00							
Frk 0.25 (% 7.18)									
Trend : 3.16364 Paralel : 4.13818		3.50							
	щł	In Made in Mar 1							
	Munu.	3.00							
		1							
the set		2.50							
1 1 m m m m m m m m m m m m m m m m m m		HAN .							
P Whyther W. W Wash		2.00							
", ' · ·									

### - Trend Kontrol - Sistem.TrendKontrol(Liste, Değer)

iDeal kullanıcıları, kodla tanımladıkları birçok trendi bir trend liste olarak tanıtırlarsa, bu listeyi tarayarak şu an bu çizdirilen trendlerin değeri nedir, trendlerden kırılan olmuş mudur gibi kontrolü Trend Kontrol fonksiyonu ile yapabilirler. Kullanım şekli aşağıdaki gibidir;

Sistem.TrendKontrol(Parametreler, out Deger)

Örnek: 2 Hisse için birer trend listesi tanımlanmıştır. Liste içerisinde her trendi tara, son barda tanımlı trendlerin değerlerini yaz, yukarı veya aşağı yönde kırılım varsa göster.

```
var TrendListesi = new List<string>();
// Trendleri Ekle SEMBOL
                               PERIYOT 1.TARIH
                                                       1.SAAT
                                                                    1.FIYAT
                                                                                 2.TARIH
2.SAAT
           2.FIYAT
TrendListesi.Add("IMKBH'GARAN ; G ; 23/02/2017 ; 00:00:00 ; 9.10 ; 04/02/2018
; 00:00:00 ; 9.20");
TrendListesi.Add("IMKBH'SAHOL ; G ; 19/05/2017 ; 00:00:00 ; 10.10 ; 01/03/2018
; 00:00:00 ; 12.20");
// Trendleri Kontrol ET
// Deger ile trendin son bardaki değeri gelir.
// Trend yukarı kestiyse Sonuc = "Y'
// Trend aşağı kestiyse Sonuc = "A"
// Trend kırılmadıysa Sonuc = ""
string Mesaj = "";
foreach (var item in TrendListesi)
{
    decimal Deger;
    var Sonuc = Sistem.TrendKontrol(item, out Deger);
    if (Sonuc != null)
    {
        Mesaj += Sonuc + " " + Deger.ToString() + "\r\n";
    }
Sistem.Mesaj(Mesaj);
```

### - Trend Score (TS) - Sistem.TrendScore(10)

iDeal indikatör kütüphanesinde yer alan Trend Score isimli indikatörü çağırır. 1 adet parametre alır ve varsayılan olarak 10 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.TrendScore(10);
Sistem.TrendScore(Liste, 10);
Sistem.TrendScore(Veriler , 10);
```

<u>ÖRNEK: +10 ile -10 arasında değer alan TS indikatörü kullanılarak MOV5/MOV22 kesilimlerinde al</u> sat yapan örnek bir strateji yazmak;

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var MOV5 = Sistem.MA(C, "Simple", 5);
var MOV22 = Sistem.MA(C, "Simple", 22);
var TS = Sistem.TrendScore(10);
var SonYon = "";
for(int i=1; i < Sistem.BarSayisi;i++)</pre>
{
   if (MOV5[i] > MOV22[i] && TS[i] == 10 && SonYon != "A")
   {
       SonYon = "A";
       Sistem.Yon[i] = "A";
   }
   else if ( MOV5[i] < MOV22[i] && TS[i] ==- 10 && SonYon != "S")
   {
       SonYon = "S";
       Sistem.Yon[i] = "S";
   }
}
```

#### - TRIX - Sistem.TRIX(12)

iDeal indikatör kütüphanesinde yer alan TRIX isimli indikatörü çağırır. 1 adet parametre alır ve varsayılan olarak 12 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

Sistem.TRIX(12);
Sistem.TRIX(Veriler, 12);

- TTI (Trend Takip İndikatörü) - Sistem.TTI(3, 2, "Simple")

iDeal indikatör kütüphanesinde yer alan TTI (Trend Takip İndikarörü) isimli indikatörü çağırır. 3 adet parametre alır ve varsayılan olarak 3/2/"Simple" kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

Sistem.TTI(3, 2, MAyontem);
Sistem.TTI(Liste, 3, 2, MAyontem);
Sistem.TTI(Veriler, 3, 2, MAyontem);

- Tipik Fiyat - Sistem.TypicalPrice()

Grafik barlarındaki (YÜKSEK + DÜŞÜK + KAPANIŞ) / 3 olarak sıkça kullanılan fiyat listesi iDealde TİPİK FİYAT olarak bir indikatör olarak çağırılabilir, çizdirilip kullanılabilir. Aşağıdaki gibi yazılır;

Sistem.TypicalPrice();

NOT: Sistem.GrafikFiyatSec("Kapanis") şeklinde grafikten veri okuyan komutta "Kapanis" yerine "Tipik" yazmak da aynı listeyi verir.

### - Ücgen Çiz - Sistem.UcgenCiz()

Grafik üzerinde kullanıcı tarafından girilen 3 noktayı (3 bar numarası ve 3 fiyat seviyesi) birleştirerek üçgen çizme amaçlı kullanılır. Özellikle patern/Flama/Formasyon bulma amaçlı kodlar yazan kullanıcılarımız, formasyonu oluşturan barları ve fiyat seviyeleri koddan tespit edip formasyonun olduğunu otomatik olarak çizdirtebilirler.

Kullanım şekli aşağıdaki gibidir;

```
Sistem.UcgenCiz(panel, bar1, fiyat1, bar2, fiyat2, bar3, fiyat3, renk, kalinlik, dolgu, dolgurenk, nokta, noktarenk)
```

#### Parantez içindeki giriş bilgileri açıklamaları aşağıdaki gibidir;

- Panel: Üçgenin grafiğin hangi paneline çizileceği girilir (barların olduğu bölge=1, alt indikatör bölgesi = 2)
- Bar1: Üçgenin birinci köşesinin denk geldiği bar numarası
- Fiyat1: Üçgenin birinci köşesinin fiyat skalasına denk gelen seviyesi
- Bar2: Üçgenin ikinci köşesinin denk geldiği bar numarası
- Fiyat2: Üçgenin ikinci köşesinin fiyat skalasına denk gelen seviyesi
- Bar3: Üçgenin üçüncü köşesinin denk geldiği bar numarası
- Fiyat3: Üçgenin üçüncü köşesinin fiyat skalasına denk gelen seviyesi
- Renk: Üçgenin dış çizgi rengi
- Kalınlık: Üçgenin dış çizgisinin kalınlık seviyesi
- Dolgu: Üçgenin içinin dolgu renkli olması isteniyorsa 1, istenmiyorsa 0 yazılır
- Dolgurenk: dolgu=1 denmişse iç dolgunun rengi verilir
- Nokta: Değer 1 yapılırsa Üçgenin köşelerinde birer nokta işareti çizilir
- Noktarenk: Köşelerde çıkan noktaların rengi belirlenir.

### ÖRNEK:

```
int SonBar=Sistem.BarSayisi;
var birincinokta = SonBar-600;
var ikincinokta = SonBar-800;
var ucuncunokta = SonBar-300;
var birincifiyatnoktasi = 154.5;
var ikincifiyatnoktasi = 190.65;
Sistem.UcgenCiz(1, birincinokta, birincifiyatnoktasi , ikincinokta, ikincifiyatnoktasi ,
ucuncunokta, ucuncufiyatnoktasi , Color.SkyBlue, 3, 1,
Color.FromArgb(100,0,0,220),0,Color.Cyan);
```





#### - Ultimate Oscillator - Sistem.UltimateOsc(7,14,28)

iDeal indikatör kütüphanesinde yer alan Ultimate Oscillator olarak bilinen indikatörü çağırır. 3 adet parametre alır ve varsayılan olarak 7/14/28 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.UltimateOsc(7, 14, 28)
Sistem.UltimateOsc(Veriler , 7, 14, 28)
```

### - VHF / Vertical Horizontal Filter - Sistem.VerticalHorizontalFilter(28)

iDeal indikatör kütüphanesinde yer alan VHF (Vertical Horizontal Filter ) olarak bilinen indikatörü çağırır. 1 adet parametre alır ve varsayılan olarak 28 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.VerticalHorizontalFilter(28)
Sistem.VerticalHorizontalFilter(Veriler, 28)
```

### - VIDYA - Sistem.VIDYA(30, 9)

iDeal indikatör kütüphanesinde yer alan VIDYA indikatörünü çağırır. 2 adet parametre alır ve varsayılan olarak 30/9 kullanılır. Aşağıdaki gibi yazım şekilleri vardır;

Sistem.VIDYA(30,9)
Sistem.VIDYA(Veriler, 30,9)

- Volume (Hacim) - Sistem.Volume()

iDeal indikatör kütüphanesindenyer alan Volume / Hacim indikatörünü çağırır.TL bazında grafiklerin altında hacim değerini çizer. Aşağıdaki şekilde kullanılır.

Not: LOT olarak çizdirmek için LOT isimli indikatör kullanılır. Ayrıca Sistem.GrafikFiyatSec("Kapanis") şeklindeki grafik barlarından veri elde eden komutta Kapanis yerine "Hacim" veya "Lot" yazarak da aynı çizgiler/indikatörler elde edilebilir.

Sistem.Volume()
Sistem.Volume(Veriler)

- Volume Oscillator Percent - Sistem.VolumeOscPercent(12,26, "Exp")

iDeal indikatör kütüphanesinde yer alan Volume Oscilatör Percent indikatörünü çağırır. 3 adet parametre alır ve varsayılan olarak 12/26/"Exp" kullanılır. Hacimdeki Yüzdesel değişimin analizini yapar. Aşağıdaki gibi yazım şekilleri vardır;

Sistem.VolumeOscPercent(12, 26,"Exp")
Sistem.VolumeOscPercent(Veriler , 12, 26,"Exp")

- Volume Oscillator Point - Sistem.VolumeOscPoint(12,26, "Exp")

iDeal indikatör kütüphanesinde yer alan Volume Oscillatör Point indikatörünü çağırır. 3 adet parametre alır ve varsayılan olarak 12/26/"Exp" kullanılır. Hacimdeki nominal değilşimin analizini yapar. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.VolumeOscPoint(12, 26,"Exp")
Sistem.VolumeOscPoint(Veriler , 12, 26,"Exp")
```

### - VORTEX - Sistem.VorteksMinus/Plus(24)

iDeal indikatör kütüphanesinde yer alan Vortex isimli indikatörü çağırır. 1 adet parametre alır ve varsayılan olarak 24 kullanılır. Tıpkı DI+ / DI- gibi "+" ve "-" diye bilinen (Plus ve Minus) iki çizgiden oluşur ve bu çizgilerin her birine ayrı bir komutla erişilebilir. Aşağıdaki gibi yazım şekilleri vardır;

```
Sistem.VortexMinus(24)
Sistem.VortexMinus(Veriler, 24)
Sistem.VortexPlus(24)
Sistem.VortexPlus(Veriler, 24)
```

ÖRNEK: Vortex çizdirmek

```
var period = 14;
var VORTEX_Plus = Sistem.VortexPlus( period );
var VORTEX_Minus = Sistem.VortexMinus(period );
```

```
Sistem.Cizgiler[0].Deger = VORTEX_Plus;
Sistem.Cizgiler[1].Deger = VORTEX_Minus;
```

- VIOP Hesap Oku - Sistem. ViopHesapOku()

Portföy penceresine ekleyerek, parola ve şifrenizi de girip LOGIN olduğunuz aracı kurumdaki hesabınızın VIOP tarafına ait bilgileri okumak için **Sistem.ViopHesapOku()** fonksiyonu kullanılır.

Sistem kod editörüne Bu komut yazılıp çalıştırıldığı zaman iDeal aracı kurumunuzdaki hesap için (VİOP TARAFI) aşağıdaki bilgileri kurumunuzdan çeker ve kullanabilmeniz için sunar;

- Teminatlar (birkaç çeşit teminat bilgisi vardır ve hepsi birer sayısal değer döner)
- Pozisyonlar (portföydeki pozisyonları içeren birden fazla elemanı olan bir listedir)
- Bekleyen Emirler (gün içi bekleyen durumunda olan emirlerinizi içeren bir listedir)
- Gerçekleşen Emirler (gün içi gerçekleşmiş emirlerinizi içeren bir listedir)

**Teminatlar** Viop Hesap Oku komutu yazıldıktan sonra, okutulan hesap dönüş listesi için kullanılan değişkenden sonra NOKTA işareti konulup aşağıdaki teminat bilgilerine erişilebilir;

```
var ViopHesap = Sistem.ViopHesapOku();
ViopHesap.TeminatToplam;
ViopHesap.TeminatBaslangic;
ViopHesap.TeminatSurdurme;
ViopHesap.TeminatKullanilabilir;
ViopHesap.TeminatCekilebilir;
```

ÖRNEK: Login olunmuş olan hesabımızdaki teminat bilgileri oku, ekrana bir tablo açıp, tabloda bu bilgileri göster formülü;

```
var ViopHesap = Sistem.ViopHesapOku();
if (ViopHesap != null) // eğer kurumdan dönen cevap boş değilse
{
   string TabloAd = "VIOP TEMINAT";
   SutunGenislik = new int[2]{200,100};
   SutunHizala = new int[2]{0,2};
   SutunBaslik = new string[2]{"Aciklama", "Değer"};
   Sistem.Tablo(TabloAd, 450, 100, 400, 200, 2, 50, SutunGenislik, SutunHizala,
SutunBaslik);
   Sistem.TabloTemizle(TabloAd);
   Sistem.TabloYazdir(TabloAd, 0, 0, "Teminat Toplamı", Color.White, Color.Black);
                                               ViopHesap.TeminatToplam.ToString("0.00"),
   Sistem.TabloYazdir(TabloAd,
                                  1,
                                         0,
Color.White, Color.Black);
   Sistem.TabloYazdir(TabloAd, 0, 1, "Başlangıç Teminatı", Color.White, Color.Black);
                                 1,
                                            ViopHesap.TeminatBaslangic.ToString("0.00"),
   Sistem.TabloYazdir(TabloAd,
                                     1,
Color.White, Color.Black);
   Sistem.TabloYazdir(TabloAd, 0, 2, "Sürdürme Teminatı", Color.White, Color.Black);
   Sistem.TabloYazdir(TabloAd,
                                1,
                                             ViopHesap.TeminatSurdurme.ToString("0.00"),
                                      2,
Color.White, Color.Black);
   Sistem.TabloYazdir(TabloAd, 0, 3, "Kullanılabilir Teminat", Color.White, Color.Black);
   Sistem.TabloYazdir(TabloAd, 1, 3, ViopHesap.TeminatKullanilabilir.ToString("0.00"),
Color.White, Color.Black);
```



Sistem.TabloYazdir(TabloAd, 0, 4, "Çekilebilir Teminat", Color.White, Color.Black); Sistem.TabloYazdir(TabloAd, 1, 4, ViopHesap.TeminatCekilebilir.ToString("0.00"), Color.White, Color.Black); }

Bu formül bir kez çalıştırılırsa (FORMÜL TEST butonuna basılırsa) aşağıdaki tablo açılır.

VIOP TEMINAT	-						
Açıklama	Değer						
Teminat Toplamı	32980.41	=					
Başlangıç Teminatı	0.00						
Sürdürme Teminatı	0.00						
Kullanılabilir Teminat	32980.41						
Çekilebilir Teminat	32980.41						

Pozisyonlar, Bekleyen ve Gerçekleşen Emirler birer Liste oldukları için, onların da dönen verilerinin her biri birden çok veri alanı sunar.

**Pozisyonlar** verisi çekildiği zaman okunabilecek bilgi alanları için, pozisyonları tanımladığımız değişkenin hemen devamına NOKTA işareti koyup sunulan bilgi alanlarının ismi yazılarak, sahip olunan hissenin adedi, maliyeti, senet kodu, kar zarar bilgisi ve Son fiyat değeri elde edilebilir. Pozisyonlar Listesinin dönüş değerleri aşağıdaki gibidir;

```
PozList.Symbol;
PozList.NetAmount;
PozList. ProfitAnlik; //anlık fiyata göre KZ
PozList.Profit; //Uzlaşıya göre KZ
```

**Gerçekleşen Emirler** verisi çekildiği zaman okunabilecek bilgi alanları için, tanımladığımız değişkenin hemen devamına NOKTA işareti koyup sunulan bilgi alanlarının ismi yazılarak, gün içinde verdiğiniz ve GERÇEKLEŞMİŞ olan emirlerinize ait fiyat, emir no, yön, emir süresi, emir tipi, emir tarihi/saati ve emir statusu (durumu) bilgileri elde edilebilir. Gerçekleşen Emirler Listesinin dönüş değerleri aşağıdaki gibidir;

```
GerceklesenList.OrderNo;
GerceklesenList.OrderDate;
GerceklesenList.Symbol;
GerceklesenList.BuySell; //Alış - Satış
GerceklesenList.Session; //süre
GerceklesenList.OrderType; //emir tipi
GerceklesenList.Price; //emir giyatı
GerceklesenList.Status; //emrin durumu
GerceklesenList.GAmount; //gerçekleşen adey
```

**Bekleyen Emirler** verisi çekildiği zaman okunabilecek bilgi alanları için, tanımladığımız değişkenin hemen devamına NOKTA işareti koyup sunulan bilgi alanlarının ismi yazılarak, gün içinde verdiğiniz ve BEKLİYOR DURUMDA OLAN olan emirlerinize ait fiyat, emir no, yön, emir süresi, emir tipi, emir tarihi/saati ve emir statusu (durumu)bilgileri elde edilebilir. Gerçekleşen Emirler Listesinin dönüş değerleri aşağıdaki gibidir;

BekleyenList.OrderNo;
BekleyenList.OrderDate;



BekleyenList.Symbol; BekleyenList.BuySell; BekleyenList.Session; BekleyenList.OrderType; BekleyenList.Price; BekleyenList.Status;

Örnek1: Hesabımı oku, VIOP tarafında gerçekleşmiş olan emirlerimi getir ve ekranda bir tablo açıp göster.

Not: Bu sonucu görebilmek için iDeal Portföy penceresinde HERSAP EKLE diyerek aracı kurumdaki hesabınızı eklemiş, parola ve/veya şifrenizi girerek hesabınıza login olmuş olmalısınız.

```
var ViopHesap = Sistem.ViopHesapOku();
if (ViopHesap != null)
{
   // Gerçekleşen Emirler
   var GerceklesenList = ViopHesap.GerceklesenEmirler;
   string TabloAd = "VIOP GERCEKLESEN"
   SutunGenislik = new int[5]{140,70,100,100,160};
   SutunHizala = new int[5]{0,1,2,2,1};
SutunBaslik = new string[5]{"Sembol", "İşlem", "Fiyat", "Miktar", "Emir No"};
Sistem.Tablo(TabloAd, 200, 500, 620, 200, 5, 100, SutunGenislik, SutunHizala, SutunBaslik);
   Sistem.TabloTemizle(TabloAd);
   for (int i = 0; i < GerceklesenList.Count; i++)</pre>
   {
       var Renk = Color.Black;
      if (GerceklesenList[i].BuySell.Substring(0,1) == "A")
          Renk = Color.Blue;
      else if (GerceklesenList[i].BuySell.Substring(0,1) == "S")
          Renk = Color.Red;
       Sistem.TabloYazdir(TabloAd, 0, i, GerceklesenList[i].Symbol, Color.White, Renk);
      Sistem.TabloYazdir(TabloAd, 1, i, GerceklesenList[i].BuySell, Color.White, Renk);
      Sistem.TabloYazdir(TabloAd, 2, i, GerceklesenList[i].Price.ToString("0.000"), Color.White,
Renk);
       Sistem.TabloYazdir(TabloAd, 3, i, GerceklesenList[i].GAmount.ToString("0"), Color.White,
Renk);
      Sistem.TabloYazdir(TabloAd, 4, i, GerceklesenList[i].OrderTime, Color.White, Renk);
   }
}
```

- Yay Çiz - Sistem.YayCiz()

Grafik üzerinde kullanıcı tarafından girilen 3 veya 4 noktayı (3/4 bar numarası ve 3/4 fiyat seviyesi) birleştirerek yay çizme amaçlı kullanılır. Özellikle patern/Flama/Formasyon bulma amaçlı kodlar yazan kullanıcılarımız, formasyonu oluşturan barları ve fiyat seviyeleri koddan tespit edip formasyonun olduğunu otomatik olarak dörtgen veya kelebek şeklinde çizdirtebilirler.

Kullanım şekilleri aşağıdaki gibidir; (3 veya 4 noktadan geçen yay olmak üzere 2 ayrı komut vardır)

Sistem.YayCiz(panel, bar1, fiyat1, bar2, fiyat2, bar3, fiyat3, renk, kalinlik, dolgu, dolgurenk, nokta, noktarenk) Sistem.YayCiz(panel, bar1, fiyat1, bar2, fiyat2, bar3, fiyat3, bar4, fiyat4, renk, kalinlik, dolgu, dolgurenk, nokta, noktarenk)

#### Parantez içindeki giriş bilgileri açıklamaları aşağıdaki gibidir;

- **Panel:** Yayın grafiğin hangi paneline çizileceği girilir (barların olduğu bölge=1, alt indikatör bölgesi = 2)
- Bar1: Yayın birinci köşesinin denk geldiği bar numarası
- Fiyat1: Yayın birinci noktasının fiyat skalasına denk gelen seviyesi
- Bar2: Yayın ikinci noktasının denk geldiği bar numarası
- Fiyat2: Yayın ikinci noktasının fiyat skalasına denk gelen seviyesi
- Bar3: Yayın üçüncü noktasının denk geldiği bar numarası
- Fiyat3: Yayın üçüncü noktasının fiyat skalasına denk gelen seviyesi
- Bar4: Yayın dördüncü noktasının denk geldiği bar numarası
- Fiyat4: Yayın dördüncü noktasının fiyat skalasına denk gelen seviyesi
- Renk: Yayın dış çizgi rengi
- Kalınlık: Yayın dış çizgisinin kalınlık seviyesi
- Dolgu: Yayın içinin dolgu renkli olması isteniyorsa 1, istenmiyorsa 0 yazılır
- Dolgurenk: dolgu=1 denmişse iç dolgunun rengi verilir
- Nokta: Değer 1 yapılırsa yayın dönüş noktalarında birer nokta işareti çizilir
- Noktarenk: Dönüş noktalarında çıkan noktaların rengi belirlenir.

#### Örnek: Dönüş noktalarının denk geldiği bar numaraları ve fiyat seviyeleri elle girilerek yay çizdirmek;

int SonBar=Sistem.BarSayisi;

Sistem.YayCiz(1, SonBar-450, 1510.3, SonBar-150, 1555.7, SonBar-100, 1610, Color.Salmon, 3, 1, Color.FromArgb(100,200,0,0),1,Color.LawnGreen);

Sistem.YayCiz4(1, SonBar-630, 1545.3, SonBar-430, 1515.3, SonBar-190, 1590.3, SonBar-10, 1530.8, Color.SkyBlue, 3, 1, Color.FromArgb(100,0,0,220),0,Color.Cyan);



### - Yazı Ekle - Sistem.YaziEkle()

iDeal sistem modülünde bir sistem/formül/indikatör yazarken, çeşitli yerlere (herhangi barın olduğu yere) bir yazı veya koddan elde edilebilen değer yazdırılmak istendiğinde bu fonksiyon kullanılır.

NOT: Bu fonksiyonun kullanım amacı, Grafik Zeminine yazı yazdırmaktan farklıdır. Zemine yazılar yazmak için kullanılan fonksiyon Sistem.ZeminYazisiEkle fonksiyonudur.

Grafiğe yazı eklenmek istediğinde, yazının yeri yatay ve dikey olarak pixel verilerek belirlenir. Bu fonksiyonda ise yazılar TEK TEK BARLAR için yazılabilir. Bu yüzden fonksiyon, yazının yazılacağı konum olarak barın hangisi olduğuna ve o barın hangi fiyat seviyesi hizasına yazılması gerektiği bilgisini de ister.

Kullanım şekli aşağıdaki gibidir;

Sistem.YaziEkle(Metin, Panel, BarNo, Fiyat, Renk, FontAdi, FontBoyutu)

#### Parantez içindeki giriş bilgileri açıklamaları aşağıdaki gibidir;

- Metin: Yazılayacak Yazı veya değer
- Panel: Yazı yazılacak panel numarası (barların olduğu bölge=1, alt indikatör bölgesi = 2, 3, 4 vs.)
- BarNo: Yazı kaç numaralı barda yazdırılacak
- **Fiyat**: Yazı o barın hangi fiyat seviyesine denk gelecek konumda olacak. (Örnek barın High değeri hizasında)
- Renk: Yazı rengi
- FontAdi: Yazının yazı tipi belirtilir. (Örnek: "Tahoma" veya "Arial")
- FontBoyutu: Yazının boyutu/büyüklüğü belirtilir. (Örnek: 16)

Örnek: 0 ve 150'lik hareketli ortalamaların kesişimlerine göre AL/SAT üreten bir sistemde, AL sinyalinin olduğu barların DÜŞÜK FİYATI seviyesine ALIŞ, SAT sinyalinin olduğu barların YÜKSEK FİYATI seviyesine SATIŞ yazdırma örneği

```
var V = Sistem.GrafikVerileri;
var C = Sistem.GrafikFiyatSec("Kapanis");
var MA1 = Sistem.MA(C, "Exp", 50);
var MA2 = Sistem.MA(C, "Exp", 150);
Sistem.KesismeTara(MA1, MA2);
var AlisRenk = Color.Cyan;
var SatisRenk = Color.Red;
for (int i = 1; i < V.Count; i++)</pre>
{
   if (Sistem.Yon[i] == "A")
   {
      var Yazi = "Alis= " + V[i].Close.ToString();
      Sistem.YaziEkle(Yazi, 1, i, V[i].Low, AlisRenk, "Tahoma", 10);
   }
   if (Sistem.Yon[i] == "S")
   {
      var Yazi = "Satis= " + V[i].Close.ToString();
      Sistem.YaziEkle(Yazi, 1, i, V[i].High, SatisRenk, "Tahoma", 10);
   }
}
```



### - Yön Listesi - Sistem.Yon()

iDeal sistem modülünde bir sistem/robot yazarken belki de en önemli ve kritik liste YÖN LİSTESİDİR. Yön, ideal sistem kütüphanesinde yer alan, kullanıcı tarafından A/S/F harfleri ile doldurularak grafiklerde sinyal üretilmesi sağlayan bir listedir. Her bir bara karşılık yön listedinde bir değer vardır ve başlangıçta bu değerler BOŞ (Yani "") durumdadır. Grafikler üzerinde AL/SAT/FLAT sinyalleri (okları) çıkmasını sağlar.

Yön listesi iki genel yöntemle oluşturulur:

1-kodu/stratejiyi yazan kullanıcılar tarafından, belirledikleri koşulları taratıp, koşul sağlandığında YÖN değerini atadıkları bir DÖNGÜ içinde doldurulur.

NOT: AL için koşul bir kez sağlanınca AL oku/sinyali işlendikten sonra, aynı koşul sonraki barlarda da tekrar sağlansa bile o barlara da yeniden AL oku/sinyali konulmaması için, koşulun içine en son yönün zaten AL OLMADIĞI kontrolü de eklenir.)

```
var RSI = Sistem.RSI(14);
var SonYon = "";
for (int i =1; i < Sistem.BarSayisi; i++)
{
    if (RSI[i] > 70 && SonYon != "A" )
    {
        Sistem.Yon[i] = "A"; // alış
        SonYon = "A";
    }
    if (RSI[i] < 30 && SonYon != "S" )
    {
        Sistem.Yon[i] = "S"; // satış
        SonYon = "S";
    }
}</pre>
```



Not: Robot kodları, sistemler gibi geçmiş barlarda ne olduğuyla ilgilenmezler. Bir ROBOT için önemli olan son barda (veya kapanmış olan son barda) EMİR GÖNDERME koşulunun olup olmadığıdır. Yani YÖN listesinin en son elemanını bilmek veya en son yönünü ne olduğunu bilmek yeterlidir. Bu yüzden, bir sistemi çağırarak emir iletimi yapan bir robot formülünde, son yönün ne olduğunu öğrenmek için Sistem.SonYonGetir komutu kullanılır.Klavuzda ilgili fonksiyonu inceleyebilirsiniz.

### - Yukarı Kestiyse - Sistem. YukariKestiyse

Bir grafik periyodunda bir çizginin bir başka çizgiyi (veya sabit bir yatay seviyeyi/sayıyı) **en son barda** yukarı kesme durumunu verir. Bu fonksiyonun dönüş değeri true veya false (yani evet veya hayır gibi) şeklindedir ve sadece son 2 bardaki değerleri kıyaslar. Geçmişten son bara kadar aşağı kesme durumlarında AL sinyali üretsin amaçlı bir formülde kullanılmaz. Genellikle Sorgu (Tarama) kodlarında kullanılır. 2 kullanım şekli vardır;

```
Sistem.YukariKestiyse(Liste1, Liste2))
Sistem.YukariKestiyse(Liste, Say1))
```

### Örnek1: Fiyatın SenkouSpanB'yi yukarı kesen senetleri tarayan sorgu kodu

```
Sistem.SorguBaslik[0] = "Fiyat";
Sistem.SorguBaslik[1] = "SenkouB";
var IchiMoku = Sistem.Ichimoku();
var SenkouB = IchiMoku.SenkouSpanB;
var C = Sistem.GrafikFiyatSec("Kapanis");
var Son = Sistem.BarSayisi-1;
if (Sistem.YukariKestiyse(C, SenkouB ))
{
Sistem.SorguDeger[0] = C[Son];
Sistem.SorguDeger[1] = SenkouB[Son];
Sistem.SorguAciklama = "Yukarı Kesti";
Sistem.SorguEkle();
}
```

### - Yüksek (Yuksek) Fiyat - Sistem.Yuksek(Sembol)

Bir sembolün o an ki fiyata göre çeşitli dönemlerde gördüğü **EN YÜKSEK FİYATI** okumak için kullanılır. Fonksiyonun içine Yüksek fiyatı okutulmak istenen sembol yazılır.

<u>Not:</u> iDeal programında bütün semboller ait oldukları piyasasının kodu ile birlikte yazılırlar. Hisse senetlerinin piyasa kodu IMKBH dır. PİYASA kodundan sonra ÜSTTEN TEK TIRNAK işareti ile ayrılıp borsadaki orijinal kod eklenir. GARAN hissesinin idealdeki sembol tanımı IMKBH'GARAN şeklindedir. Örneğin USDTYR için FX'USDTRY şeklinde yazılır. Bir sembolün PİYASA kodunun ne olduğu, o sembolü sayfanıza yazarken @ işareti yanında gösterilir.)

Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.YuksekAltiAy(Sembol);
Sistem.YuksekBirAy(Sembol);
Sistem.YuksekBirHafta(Sembol);
Sistem.YuksekBirYil(Sembol);
Sistem.YuksekBuAy(Sembol);
Sistem.YuksekBuHafta(Sembol);
Sistem.YuksekBuYil(Sembol);
Sistem.YuksekGun(Sembol);
Sistem.YuksekSeans(Sembol);
Sistem.YuksekUCAy(Sembol);
```

Örnek: GARAN hisse senedinin, son 6 ayın en yüksek değerinin okuyup ekrana mesaj olarak çıkaran kod:

```
var AltiAyYuksek = Sistem.YuksekAltiAy(Sembol);
Sistem.Mesaj("6 Aylık en yüksek Fiyat = " + AltiAyYuksek.ToString());
```

### - Yüzde Değişim - Sistem.Yuzde(Sembol)

Bir sembolün o an ki fiyata göre çeşitli dönemler için YÜZDE DEĞİŞİM bilgisini okumak için kullanılır. Fonksiyonun içine Yüzde Değişim fiyatı okutulmak istenen sembol yazılır.

Kullanım şekilleri aşağıdaki gibidir.

```
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
Sistem.YuzdeAltiAy(Sembol);
```

- Yüzeysel Liste Getir - Sistem.YuzeyselListeGetir(Kriter)

iDeal Veri Terminalinde, sayfalarımızda izlediğimiz fiyat pencerelerinde sütun başlıklarına atadığımız bütün sütunlara ait verileri bütün semboller için tek seferda okutabileceğimiz bir fonlksiyondur. Bütün semboller birer ÖN EK / PREFIX ile tanımlanır idealde ve sayfamıza bir kod eklerken klavye ile yazım yaptığımız sırada bize bu prefix'in ne olduğunu gösterir. Kullanım şekli aşağıdaki gibiri;



#### Sistem.YuzeyselListeGetir(kriter)

Aşağıdaki fotoda sayfamıza bir kod eklemek üzereyken, yazdığımız harf kadarıyla uyuşan semboller ve o sembollerin hangi piyasaya ait olduğu yanlarında @ işareti ile gösterilmektedir.

Hisse Senetleri ve Varantlar : IMKBH Borsa İstanbul Endeksleri: IMKBX VIOP Vadeli sözleşmeler ve Opsiyonlar: VIP Çapraz Kurlar/Pariteler : FX Kıymetli Metaller: KIYM KapalıÇarşı Serbest Piyasa Verileri: SERPIY Liborlar: LIBOR Dünya Bonoları: WBOND Çeştili Piyasalar: DFN



Eğer bütün Hisse Senetleri Piyasasına ait BÜTÜN bilgileri tek seferde okutmak istersek IMKBH için Yüzeysel Liste Getir komutu kullanırız ve aşağıdaki sayfada yer alan (sayfamıza konulabilecek bütün sütun başlıklarındaki bütün verileri içeren) bir liste elde ederiz.

IDEAU	GO	BIST	VIOF	Fai	z Diğer		-	· 🛃 💾		GARAN		🔄 📄 🌈	≞	🕂 🛅 🛸	<b>_</b>	<u></u>
	L VIIIOO															
Kod	Son Firk	Sopla	ot Vä		ALEXT	Allot	Call Fuel	SatLat	Miss	Delt	ör	elf Erb	<b>1</b> 4 4			- <b>-</b>
KUU	sonaryt	SUIL		_	матус	ALLOU	Sataryt	Salifor	TRS	Dşk						•
AEFES	23.	.56	2265		23.56	4943	23.58	1258		23.80	23.44	23.58	0.18	Ы	-	
AGHOL	21.	51	323	1	27.50	2400	21.32	250669		20.30	5.40	20.00	-0.50	R. i. l. et		
AKONK	16	85	66	i.	16.85	102527	16.86	5000		18 32	16.75	16.87	-0.04	B.A.LUL B.A.O	<b>^</b>	
AKGRT	9	32	380	1	9.31	389	9.32	6644		9.41	9.13	9.25	0.07	BAOrt		
AKSA	13.	.43	45		13.43	17716	13.44	1033		13.74	13.01	13.32	0.11	B.S.Lot		
AKSEN	7.	.17	25	1	7.16	1500	7.17	264283		7.26	7.13	7.18	-0.01	B.S.O		•
AKSGY	3.	56	5000	1	3.55	65725	3.56	i 15927	,	3.59	3.53	3.56	0.00	B.S.Ort		
ALARK	7.	.80	126	1	7.80	146181	7.81	l 65000	1	7.89	7.56	7.57	0.23	Bariver	=	ır
ALBRK	2.	.02	2500	÷	2.01	249619	2.02	2225050	1	2.05	1.99	2.04	-0.02	Başl.Trh		
ALCTL			1017		29.26	633	29.48	3 847		30.54	29.26	29.88	-0.62	Baz		
ALGYO	23.	.90	1	+	23.88	2764	23.90	) 190		25.16	23.14	24.10	-0.20	Borsa		
ALKIM	14.	.44	500	1 1 N	14.44	4419	14.45	5 3698		14.62	14.39	1:40	0.04	Brut Takas		
ARCLK	30.	.00	10	1	30.00	25556	30.04	i 150		30.32	29.90	30.22	-0.22	Coupon		
ARDYZ	45.	.62	21	1	45.62	7742	45.66	621		47.30	45.52	46.86	-1.24	Currency		
ASELS	18.	.00	100	1	18.00	38991	18.01	L 60114	•	18.18	17.95	18.02	0.02	CY		
AYGAZ	14.	.30	2500	1	14.30	22625	i 14.36	i 155		14.50	14.28	14.57	-0.07	Çarpan		
BAGFS	19.	.93	6	1	19.92	3787	19.93	3 7521		20.46	19.87	20.38	-0.45	Dayanak		
BIMAS	73.	.20	20	•	73.15	172	73.20	) 11240	)	73.65	72.95	73.20	0.00	Def.Değ		
BIZIM	15.	.53	13	1	15.52	5148	15.5	12837		15.75	15.48	15.65	-0.12	Dng.Al.K		
BRISA	19.	.89	25	•	19.86	10524	19.89	1110	1	20.32	19.82	20.00	-0.11	Dng.Frk		
BRSAN	30.	.30	10	. <u>†</u>			30.30	971839		36.50	30.30	33.66	-3.36	Dng.Frk%		
BUCIM	10.	.85	4	1	10.84	19691	10.85	5 1831		11.18	9.75	10.17	0.68	Dng.Fyt		
CCOLA	61.	.80	14	1	61.80	1662	62.00	1 41147	-	63.20	61.60	62.60	-0.80	Dng.Lot		
														Dng.Lot.Frk		
														Ding.sat.k		
														Donem		
														Dişk Dişk ûlu		
														Disk Aud		
														Disk Avia		
														Disk Ave	-	
														D J K MYO		

Kullanım Detaylarını bir örnekle aşamalandıralım;

Bütün hisse senetleri piyasası yüzeysel verilerini çekelim ve "Liste" isimli bir değişkene atayalım;

```
var Liste = Sistem.YuzeyselListeGetir("IMKBH'");
```

Şimdi bu listede sahiğp olduğumuz verilerin neler olduğuna nerden bakacağız? Kod yazma penceremiz olan SİSTEM TANIMLARI penceresinin, orta/sağ kısmındaki YÜZEYSEL isimli kutuda yer alan tüm bilgi alanları bu fonksiyon yazıldığı anda elimizdedir ve listeyi atadığımız değişken isminin yanına nokta koyarak bu kutudaki bilgilere erişebilir.



Burada veri alanları İngilizce kavramlar olarak sunulmuştur. Bazı çok kullanılan bilgi alanları için hangi kelime/komut yazılmıştır aşağıda açıklanmıştır.

Son Fiyat: LastPrice Alış Fiyatı: BidPrice Satış Fiyatı: AskPrice Tavan Fiyatı: LimitUp Taban Fiyatı: LimitDown Yüzde Değişim: NetPerDay Pazar Bilgisi: Grup (A=Ana Pazar, Y=Yıldız Pazar, ALT= ALT Pazar) Seri Kodu: Seri Brüt Takas Kapsamı: FI182 (=1 ise brüt takas kapsamında, =0 ise değil) Endeks 30/50/100 kapsamı: IndexType (100= xu100, 110= xu050, 111=xu030)

**NOT:** Bu komut bir piyasa için BÜTÜN SEMBOLLERE ait verileri getirir. Bir tek sembol için ayrı bir komut vardır (YüzeyselVeriOku) ve o klavuzun bir sonraki maddesinde açıklanacaktır. Bür çok sembol için bir liste getirdiği için, listedenin içinde bir döngü ile dolaşarak her sembolün bilgileri okutulur.

ÖRNEK: Bütün hisselerin yüzeysel veri listesini oku, XU100 kapsamında olan hisseleri seç, hepsinin yüzde değişimlerini oku, yüzde değişimi pozitif olanları ayrı, negatif olanları ayrı topla. Pozitif yüzde değişimler toplanını negtif yüzde değişimler toplanına böl ve sonucu mesajla ekranda göster.

```
var Liste = Sistem.YuzeyselListeGetir("IMKBH'");
string Mesaj = "";
var artanlar = 0.0;
var azalanlar = 0.0;
var oran = 0.0;
for (var i = 0; i < Liste.Count; i++)</pre>
{
   if (Liste[i].IndexType == "100" )
   {
      var Sembol = Liste[i].Symbol;
      var Veri = Sistem.YuzeyselVeriOku(Sembol);
      var Degisim = Veri.NetPerDay;
      if (Degisim < 0)</pre>
         azalanlar++;
      else if (Degisim > 0)
         artanlar++;
      if (azalanlar != 0)
         oran = artanlar / azalanlar ;
         Mesaj = oran.ToString("0.00");
         Sistem.Mesaj(Mesaj);
      }
}
```

ÖRNEK2: Yıldız ve Ana Pazarda olan, Bürt Takas Kapsamında olmayan, fiyatı 100 liradan küçük hisseleri bir listeye kaydet;

```
var Senetlist = new List<string>();
var Liste = Sistem.YuzeyselListeGetir("IMKBH'");
for (int i = 0; i < Liste.Count; i++)
{
    if ((Liste[i].Grup == "Y" || Liste[i].Grup == "A" ) && Liste[i].FI182 != 1 &&
Liste[i].LastPrice < 100 && Liste[i].Seri == "E" )
        Senetlist.Add(Liste[i].Symbol);
}
```

### - Yüzeysel Veri Oku - Sistem.YuzeyselVeriOku(Sembol)

Bir üst bölümde anlatılan yüzeysel veri setinin bir tek sembol için olan fonksiyonudur. Geri kalan bilgi alanları tamamen aynıdır. Kullanım şekli aşağıdaki gibidir;

```
Sistem.YuzeyselVeriOku(Sembol)
```

#### ÖRNEK: Vadeli 30 endeks kontratının açık pozisyon sayısı verisine ulaşmak

```
var Sembol = "VIP'VIP-X030";
var V = Sistem.YuzeyselVeriOku(Sembol);
var APS = V.OpenInterest;
```

```
Sistem.Mesaj(APS.ToString());
```

ÖRNEK2: Bir hisse bazı yüseysel verilerini okutup ekrana mesaj olarak çıkaraım;

```
var Sembol = "IMKBH'PGSUS";
var Veriler = Sistem.YuzeyselVeriOku(Sembol);
var Last = Veriler.LastPrice;
var Open = Veriler.OpenDay;
var Bid = Veriler.BidPrice;
var Ask = Veriler.AskPrice;
var Tavan = Veriler.LimitUp;
var Taban = Veriler.LimitDown;
var PrevClose = Veriler.PrevCloseDay;
var Saat = Sistem.Saat;
var AortH = Veriler.Wavr2Week1;
```

```
Sistem.Mesaj(Sembol
+ "\r\n" + "Yayın Saati =" + " " + Saat.ToString()
+ "\r\n" + "Son Fiyat =" + " " + Last.ToString()
+ "\r\n" + "Açılış Fiyatı =" + " " + Open.ToString()
+ "\r\n" + "Alış =" + " " + Bid.ToString()
+ "\r\n" + "Satış =" + " " + Ask.ToString()
+ "\r\n" + "Tavan =" + " " + Tavan.ToString()
+ "\r\n" + "Taban =" + " " + Taban.ToString()
+ "\r\n" + "Ağırlıklı Ort =" + " " + AortH .ToString()
+ "\r\n" + "Önceki Kap. =" + " " + PrevClose .ToString());
```

#### YuzeyselVerileriOku

```
IMKBH'PGSUS
Yayın Saati = 17:25:27
Son Fiyat = 71.9
Açılış Fiyatı = 73
Alış = 71.9
Satış = 71.95
Tavan = 79.05
Taban = 64.75
Ağırlıklı Ort = 70.78339
Önceki Kap. = 72.3
```

- Zemin Yazısı Ekle - Sistem.ZeminYazisiEkle()

SEZAİ KILIÇ

iDeal Sistem modülünde bir formül yazarken, çeşitli durumlarda grafik ekranı zeminine yazılar yazmak istendiğinde bu fonksiyon kullanılır. Zemine yazı eklenmek istendiğinde, yazının yeri (yatay ve dikey olarak pixel cinsinden), panel numarası (grafik verilerinin olduğu yer veya alttaki indikatör panelleri), rengi, font adı ve boyutu belirtilmelidir.

Kullanım şekli aşağıdaki gibidir;

Sistem.ZeminYazisiEkle(Metin, Panel, X, Y, Renk, FontAdi, FontBoyutu)

Sistem.YaziEkle fonksiyonu, mutlaka belirtilmesi gereken 7 adet parametreye ihtiyaç duyar. Bunlar aşağıdadır.

Metin: (Çift tırnak içinde, yazdırılmak istenen metin)

Panel: (grafik panel numarası1'dir. İndikatör panelleri de 2'den başlar ve devam eder)
X: (Yazının başlayacağı nokta, grafik zemininin EN SOL kenarından kaç pixel uzak olsun.? Sola bitişik yazı için X değeri SIFIRDIR.

Y: (Yazının bulunduğu dikey seviye, grafiğin en üst noktasından kaç pixel aşağıda olsun.? Üst/Tepe noktaya yapışık yazı için Y değeri SIFIRDIR.

**Renk:** (Sistem.Renk fonksiyonuyla veya Color.Red şeklinde tanımlanır.)

FontAdi: (Çift tırnak içinde, Windows'a tanımlı bir yazı tipi adı girilir. (Örn: "Tahoma") FontBoyutu: (Tamsayı olarak, yazının font büyüklüğü belirtilir.

Bu fonksiyon çok çeşitli amaçlar için kullanılabilir.

- Sistem, indikatör vs yazmıyor olsanız bile, teknik analiz çalışmalarınıza isim/unvan/logo/reklam/slogan veya notlar yazdırmak amacıyla kullanabilirsiniz.
- Teknik analiz yaparken, her an gözünüzün önünde durmasını istediğiniz bazı verileri (başka bir kodun son fiyatı, grafikte atılı olmayan bir indikatörün herhangi bir bardaki değeri, bilanço/yüzeysel/derinlik verileri veya kendi hesaplattığınız herhangi bir bilgiyi zemine yazdırabilirsiniz
- AL/SAT stratejisi olan bir sisteminiz varsa, sisteminizin yönüne bağlı olarak değişik yazı/not/veri yazdırtabilirsiniz
- AL/SAT stratejisi olmasa dahi, başka koşullar tanımlayıp, her bir koşul için başka yazı/not veya veri zemine yazdırılabilir. (Sistem ALDA ise xxxx, SATTA ise yyy, FLAT durumda (pozisyonsuz) ise xxx yazdırmak gib

### ÖRNEK1: Takip ettiğimiz hisse grafiği üzerinde Dolar ve endeks fiyatını, ayrıca hissenin tahtasındaki verileri zeminde yazı olarak görmek;

```
var Sembol1 = "FX'USDTRY";
var Derinlik = Sistem.DerinlikVerisiOku(Sistem.Sembol);
var Alis = Derinlik.Bids[0].Price;
var ALot = Derinlik.Bids[0].OrderCount;
var Satis = Derinlik.Asks[0].Price;
var SLot = Derinlik.Asks[0].Size;
var SEmir = Derinlik.Asks[0].OrderCount;
Sistem.ZeminYazisiEkle("Derinlik 1.Kademe Bilgileri", 1, 450, 20, Color.Cyan, "Tahoma", 13);
Sistem.ZeminYazisiEkle("XU100 = " + " " + Sistem.SonFiyat("IMKBX'XU100"), 1, 160, 80, Color.Gray,
"Tahoma", 15);
```

Sistem.ZeminYazisiEkle("USDTRY= " + " " + Sistem.SonFiyat(Sembol1), 1, 160, 100, Color.Gray, "Tahoma", 15); Sistem.ZeminYazisiEkle("Alış = " + " " + Alis.ToString(),1, 400, 60, Color.Gold, "Tahoma", 12); Sistem.ZeminYazisiEkle("Alış Lot = " + " " + ALot.ToString(),1, 400, 80, Color.Gold, "Tahoma", 12); Sistem.ZeminYazisiEkle("Alış Emir= " + " " + AEmir.ToString(),1, 400, 100, Color.Gold, "Tahoma", 12); Sistem.ZeminYazisiEkle("Satış = " + " " + Satis.ToString(),1, 600, 60, Color.Gold, "Tahoma", 12); Sistem.ZeminYazisiEkle("Satış = " + " " + Satis.ToString(),1, 600, 60, Color.Gold, "Tahoma", 12); Sistem.ZeminYazisiEkle("Satış Lot = " + " " + SLot.ToString(),1, 600, 80, Color.Gold, "Tahoma", 12); Sistem.ZeminYazisiEkle("Satış Emir= " + " " + SEmir.ToString(),1, 600, 100, Color.Gold, "Tahoma", 12); Sistem.ZeminYazisiEkle("Satış Emir= " + " " + SEmir.ToString(),1, 600, 100, Color.Gold, "Tahoma", 12);



### - Zaman Konrol / Güncelle - Sistem.ZamanKontrol()

iDeal sistem modülünde, PERİYODİK ARALIKLARLA bir kontrol, hesap ve işlem yaptırtmak isteyen kullanıcılar Zaman Kontrol fonksiyonlarını kullanır. Saniye veya Dakika bazında iki ayrı zaman kontrol fonksiyonu vardır. Bu fonksiyonlardan biri kullanıldığında, periyodun güncellenip, yeni bir tura devam edilmesi için ZamanKontrolGuncelle fonksiyonu da bunlarla birlikte kullanılmalıdır.

ÖRNEK: IDEAL'in gelişmiş sistem modülü, aynı kod içinde birden farklı zaman aralıklarında, birden farklı iş yapmaya da izin verir. Bu nedenle, söz konusu fonksiyonların bir anahtar değeri (parantez içinde sembol adı veya bir anahtar sözcük yazılması) gerekir.

Örneğin, her 30 saniyede bir ekranın fotosunu çek, her 50 dakikada bir çekilmiş son fotoyu mail olarak gönder gibi bir komut içeren bir robot kodu yazabilirsiniz.

Saniye ve Dakika bazında sunulan fonksiyonların ikisini de ayrı ayrı aynı kod içinde kullanmaya imkân verilir. Bu durumda fonksiyonların parantezlerinin içine anahtar kelime olarak sizce anlamlı birer değer verebilirsiniz.

```
if (Sistem.ZamanKontrolSaniye("FOTOCEK") >= 30)
```

SEZAİ KILIÇ

{

```
Sistem.ZamanKontrolGuncelle("FOTOCEK");
Sistem.GoruntuKaydet("C:\\test.png");
}
if (Sistem.ZamanKontrolDakika("MAILAT") >= 50)
{
Sistem.ZamanKontrolGuncelle("MAILAT");
Sistem.GoruntuKaydet("C:\\test.png");
//MAIL GÖNDER
}
```

ÖRNEK2: Her 10 saniye aralıklarla 1 lot DOHOL alım emri gönder

```
var Sembol = "IMKBH'DOHOL";
var Miktar = 1;
if (Sistem.ZamanKontrolSaniye(Sembol) >= 10)
{
    Sistem.ZamanKontrolGuncelle(Sembol);
    Sistem.EmirSembol = Sembol;
    Sistem.EmirSlem = "Alış";
    Sistem.EmirIslem = "Alış";
    Sistem.EmirIslem = "Alış";
    Sistem.EmirIslem = "Alış";
    Sistem.EmirIslem = "Alış";
    Sistem.EmirIslem = "Alış";
    Sistem.EmirIslem = "KIE";
    Sistem.EmirSuresi = "KIE";
    Sistem.EmirGonder();
}
```

```
- ZigZag / Peak-Through - Sistem.Zigzag()
```

Zizgzag isimli indikatörü çağırıp hesaplamak için birden çok iDeal fonksiyonu bulunmaktadır. Bunların açıklamaları ve kullanım detayları aşağıda belirtilecektir. Fakat çok önemli bir detay olarak, ZİG ZAG indikatörünün RE-PAINT yapan (barlar ilerledikçe geçmişdeki değerini yeniden hesaplayıp değiştiren) bir indikatördür. Sistemlerde kullanıldığı zaman çok yüksek getiriler gösterilir. Bu performans sonuçları aldatıcıdır ve sistemlerde kullanılması asla tavsiye edilmez.

ZigZag için toplam 7 adet ideal fonksiyonu bulunmaktadır. Bunların yazım şekilleri aşağıdadır. Percen olanlar YÜZDE kullanır, POİNT olan komutlar puan/tl değişimi kullanır.

```
Sistem.ZigZagPercent(5)
Sistem.ZigZagPercent(Liste, 5)
Sistem.ZigZagPercent(Veriler, 5)
Sistem.ZigZagPoint(1)
Sistem.ZigZagPoint(Liste, 1)
Sistem.ZigZagPoint(Veriler, 1)
Sistem.ZigZagPeakThrough(Liste,Sayi, Yuzde)
```

ZigZag indikatörü için Dip/Tepe dönüllerinden beri geöen bar sayıları veya Dip ve Tepenin hangi seviye olduğunu doğrudan rişim için 4 ayrı liste/indikatör çıktısı daha vadır. Bunlara erişebilmek için ZigZagPeakThrough fonksiyonu kulanılır. Dönüş 4 ayrı LİSTE/ÇİZGİ içeren bir listedir ve aşağıdaki şekilde bu alt listelere erişiriz;

```
var C = Sistem.GrafikFiyatSec("Kapanis");
var Z = Sistem.ZigZagPeakThrough(C,1, 1);
var Peak = Z.Peak;
var Through = Z.Through;
var PeakBars = Z.PeakBars;
var ThroughBars = Z.ThroughBars;
Sistem.Cizgiler[0].Deger = Peak; //Panel1
```



Sistem.Cizgiler[1].Deger = Through; //Panel1
Sistem.Cizgiler[2].Deger = PeakBars; //Panel2
Sistem.Cizgiler[3].Deger = ThroughBars; //Panel3



### - Zirve - Sistem. Zirve(Barsayısı)

Belli bir bar sayısı kadar bölgenin Zirve seviyesini gösteren ZİRVE indikatörünü hesaplayan fonksiyondur. Girilen bar sayısı kadar öncesi ve sonrası bölgenin en zirve noktasını verir. Kullanım şekli aşağıdaki gibidir;

Sistem.Zirve(100);

Bkz: Sistem.Dip(100) aynı zamanda.

Örnek1: 50 periyotluk Dip/Zirve indikatörü çizdirme kodu

```
var Dip = Sistem.Dip(50);
var Zirve = Sistem.Zirve(50);
Sistem.Cizgiler[0].Deger = Dip;
Sistem.Cizgiler[1].Deger = Zirve;
```



Yazan: Sezai KILIÇ @2020