کارگاه معاملات الگوریتمی

QuantoRythm

آیکو
شرکت مشاور سرمایه گذاری
ارزش پرداز آریان
تحت نظارت سازمان بورس و اوراق بهادار

ارائه:
مهدی سلیمانی
Mahdi.soleymani@gmail.com

زمستان ۹۷

بخش های اصلی

- بخش اول: تعریف و جایگاه معاملات الگوریتمی
- بخش دوم: بررسی امکانات و ویژگی های سامانه QuantoRythm
- بخش سوم: آشنایی با زبان C# و نوشتن الگوریتم

تعریف معاملات الگوریتمی

به فرآیند به کارگیری کامپیوتر در انجام معاملات با استفاده از برنامه نویسی معاملات الگوریتمی گفته میشود. در این نوع معاملات یک سیستم نرم افزاری برروی پارامترهایی مانند زمان ارسال، قیمت یا حجم سفارشات تصمیم میگیرند.

مزایای معاملات الگوریتمی

- سرعت و دقت بالا در ارسال سفارشات
- قابلیت رصد و پیگیری وضعیت بازار برای سهم های مختلف و تصمیم گیری در زمان مناسب برای ارسال سفارش
- کاهش خطا های انسانی که ممکن است بر اثر هیجانات و فاکتورهای فیزیولوژیکی رخ دهد.
- قابلیت بک تست الگوریتم و استراتژی روی داده های تاریخی و واقعی

فرایند انجام معاملات

- احراز هویت
- مدیریت و کنترل قدرت خرید
- ارسال سفارشات به هسته معاملات

کارگزاری

سرمایه گذار

هسته معاملات

نیازمندی های کار با سامانه‌های معاملات الگوریتمی

- شناخت بازار بورس و نحوه انجام معاملات
- آشنایی با تحلیل داده ها و مدل های آماری
- توانایی کار با یک زبان برنامه نویسی

QuantoRythm

- ارائه زیر ساخت تعریف و اجرای الگوریتم
- ساده سازی و تسهیل نوشتن الگوریتم
- گزارش رفتار و بازدهی الگوریتم
- امکان دسترسی و مدیریت همزمان چندین حساب کارگزاری
- امکان ارسال سفارشات به طور همزمان از طریق کارگزاری های مختلف
- امکان Back Test الگوریتم ها

معرفی نرم افزار

اطلاعات بازار

شاخص‌های تحلیل تکنیکال

QuntoRythm

اطلاعات پرتفولیو

اطلاعات مالی

وضعیت سفارشات

اطلاعات بازار

- Pclosing قیمت پایانی
- TradesCount تعداد معاملات
- TradesVol حجم معاملات
- TradesValue ارزش معاملات
- NAV خالص ارزش دارایی‌ها
- LastTradeDateTime زمان اخرین معامله
- BestSellers بهترین فروشندگان
- BestBuyers بهترین خریداران

اطلاعات مالی

- Withdrawable مبلغ قابل برداشت
- BalanceBlocked مبلغ بلاک شده
- TotalBalance مانده حسابداری

اطلاعات پورتفولیو

- Symbol نماد
- Quantity تعداد

اطلاعات سفارشات باز

- InstrumentId کد سهم
- OrderDateTime زمان سفارش
- Price قیمت
- OrderIdOms کد سمت بورس
- OrderSide خرید/فروش
- QuantityExecuted تعداد اجرا شده
- OrderStatus وضعیت سفارش

اطلاعات معاملات

Vol	حجم معاملات
TradeTime	زمان انجام معامله
OrderSide	خرید/ فروش
Price	نرخ

شاخص‌های تحلیل تکنیکال

توابعی هستند که بر روی گروهی از داده ها اعمال میشوند و خروجی هایی را تولید میکنند

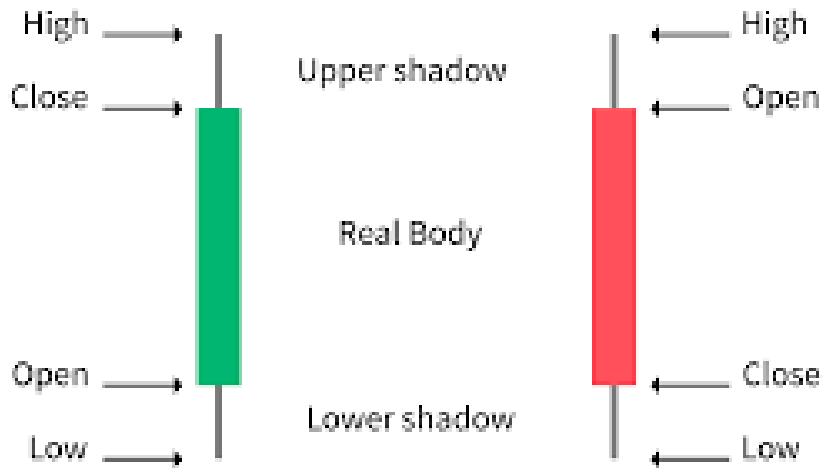
چند نمونه از توابع قابل استفاده

- RSI
- MACD
- MovingAveragre

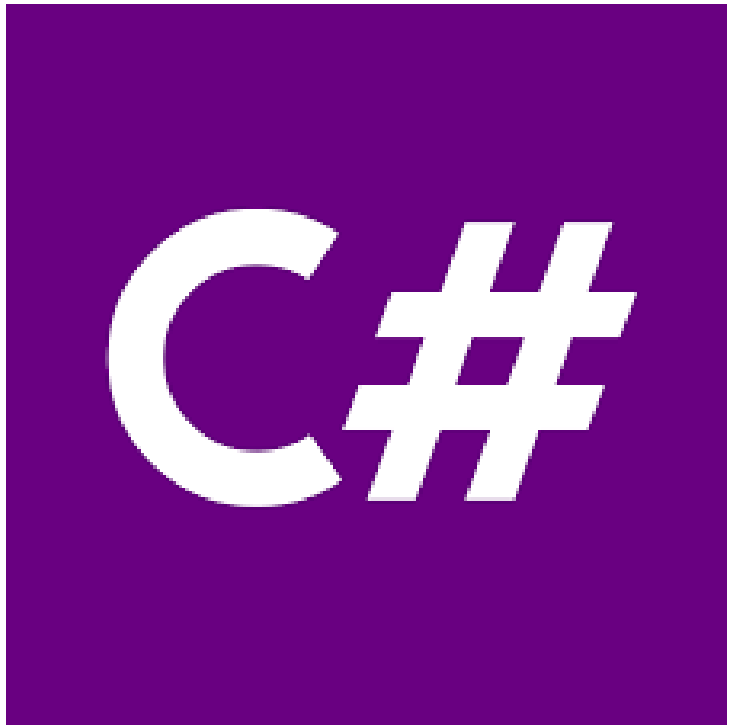
مفهوم کندل

هر کندل وضعیت یک سهم را در یک برش زمانی نمایش میدهد

- Timeframe
- Period
- AppliedPrice



IcoSMA(“IRO3ZOBZ0001”,TimeFrameEnum.D1,50,AppliedPriceEnum.Close)


C#

نمونه یک کد معاملات الگوریتمی

```
var omsData = GetOmsData("123_12");
var myAccount= GetAccount("123_12");
var marketdata = GetMarketData("IRO3ZOBZ0001");

var change= marketdata.PClosing / marketdata.PriceYesterday;
var hasPendingOrder= omsData.HasPendingOrderFor("IRO3ZOBZ0001");
var hasOpenorder = omsData.HasOpenOrderFor("IRO3ZOBZ0001");
var totalExecuted = omsData.SumExecutedTradeFor("IRO3ZOBZ0001");

var buyerPrice = marketdata.BestBuyers.Qoutes[0].Price;
var myBuyPrice = buyerPrice - 1;

var qtyForFirstAndSecondRow = marketdata.BestSellers.Qoutes[0].Quantity + marketdata.BestSellers.Qoutes[1].Quantity;
var totalTrades = marketdata.TradesVol;

var sma = IcoSMA("IRO3ZOBZ0001", TimeFrameEnum.D1, 20, AppliedPriceEnum.Close);
var amount =Math.Round(omsData.CreditInfo.Withdrawable * 0.8) ;
int qty =(int)Math.Round( amount / myBuyPrice);

if ( !hasOpenorder && !hasPendingOrder  && totalExecuted<2500 &&
    marketdata.PClosing >1180 &&
    totalTrades>1500000000 &&
    qtyForFirstAndSecondRow>320000000 &&
    sma==1800
      )
{
    this.SendMyOrderNow(myBuyPrice, qty, "IRO3ZOBZ0001", myAccount, OrderSide.Sell);
}
```

مفاهیم

- متغیر
- عملگرهای ریاضی
- عملگرهای منطقی
- دستورات شرطی
- حلقه های تکرار
- آرایه
- تابع
- Enum
- مفهوم کلاس در شی گرایی
- وراثت
- Overriding

متغیر

محلی است در حافظه برای نگهداری دیتا

نحوه تعریف متغییر نوع نام;

int	عدد صحیح		string	رشته ای
double	عدد اعشاری		boolean	منطقی
char	کاراکتر			

```
int x1;      double f;   string symbolName;

x1 = 12;      f = 23.123;    symbolName = "خودرو";
```

عملگرهای ریاضی

+	جمع		--	کاهش یک واحد
-	تفریق		+=	جمع و انتصاب
++	افزایش یک واحد		-=	تفریق و انتصاب
/	تقسیم		*	ضرب
%	باقی مانده			

```
int x1,x2;
x1=12;
x2 = 10;
x1++;
x1 += x2;
```

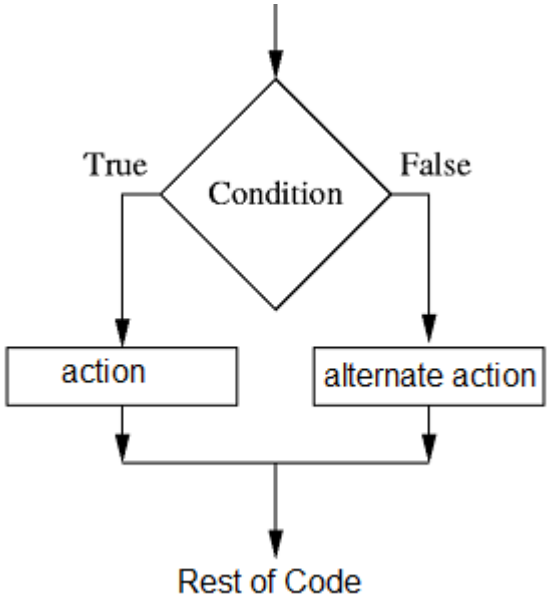
انواع عملگرهای منطقی

<	کوچکتر		!=	نا مساوی
>	بزرگتر		==	مساوی
<=	کوچکتر و مساوی		&&	And
>=	بزرگتر و مساوی		||	Or

```
bool isGoodTimeForBuy;
int closingProice = 2500;

isGoodTimeForBuy = closingProice > 3000;

Console.Write(isGoodTimeForBuy);
```

دستورات شرطی

```
if ( عبارت منطقی )
{
    SomeAction
}
else
{
    SomeAction
}
```



دستورات شرطی

```
if (x == 1)
{  //DO Action
}
else
{  if (x == 2)
    { //DO Action
    }
    else
    {  if (x == 3)
        {  //DO Action
        }
        else
        {  if (x == 4)
                { //DO Action
                }
        }
    }
}
```

```
switch ( x )
{
    case 1:  یک یا چند دستور  ; break;
    case 2:  یک یا چند دستور  ; break;
    case 3:  یک یا چند دستور  ; break;
    case 4:  یک یا چند دستور  ; break;
    default:
                در صورتی که هیچ کدام از شرط ها برقرار نبود
        break;
}
```

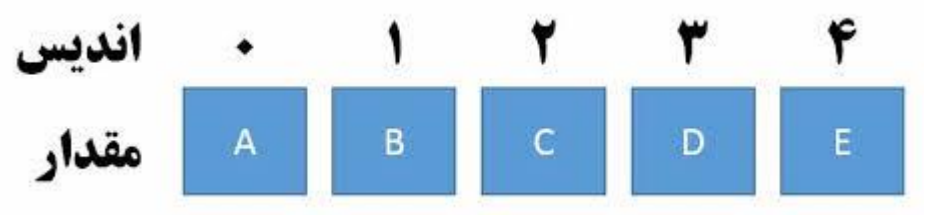
دستورات تکرار

```
for (int i = 0; i < 10; i++)
{
    //Do Action
}
```

```
while ( عبارت منطقی )
{
    //Do Actions;
}
```

آرایه

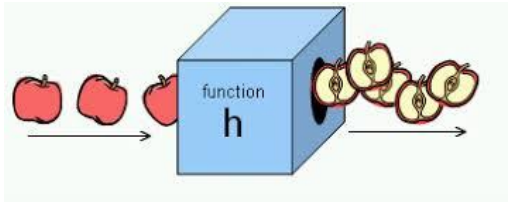
آرایه مجموعه‌ای از خانه های حافظه که هم نوع هستند و در مجاور هم قرار دارند و از طریق یک نام قابل دسترسی هستند.



مثال: محاسبه میانگین اعداد در یک آرایه

```
int[] closingPriceList = new[] { 2500, 2600, 2700, 2600 };

int sum = 0;

for (int i = 0; i < closingPriceList.Length; i++)
{
    sum += closingPriceList[i];
}

int avg = sum/4;
```

تعریف تابع (متد)




```
public int Avg(int HighPrice,int lowPrice)
{
    return (HighPrice + lowPrice) / 2;
}
```

```
public void ShowMessage(string message)
{
    //print message
}
```

```
int x = Avg(100, 150);

ShowMessage("Hi");
```

Enum

Enum در واقع یک سری عدد صحیح ثابت constant integer است که هر یک موقعیت خاصی در لیست دارد و با اسم مشخصی شناسایی می شود

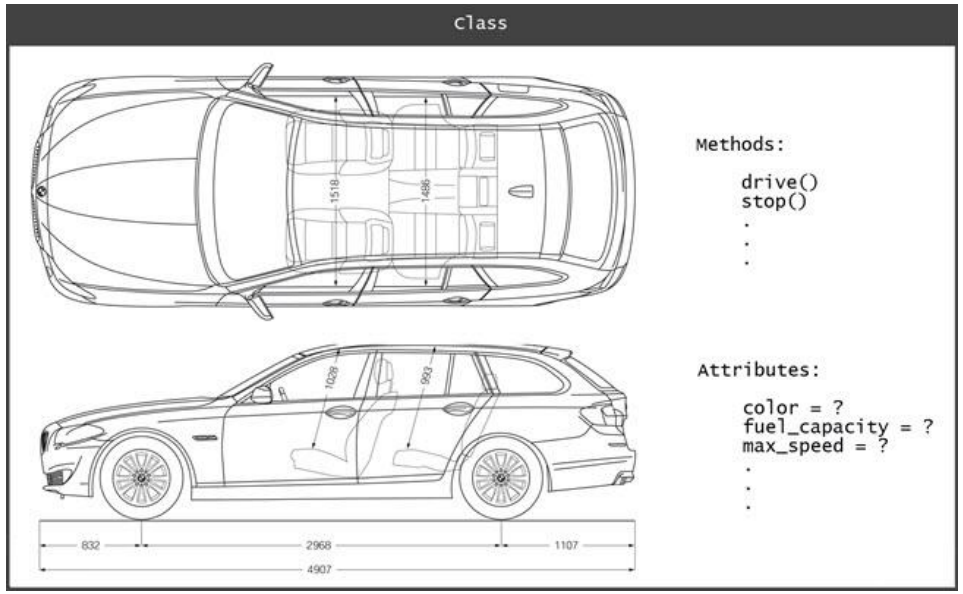
```
Public enum Colors {
Red=1,
Green=2,
Blue=3
}
```

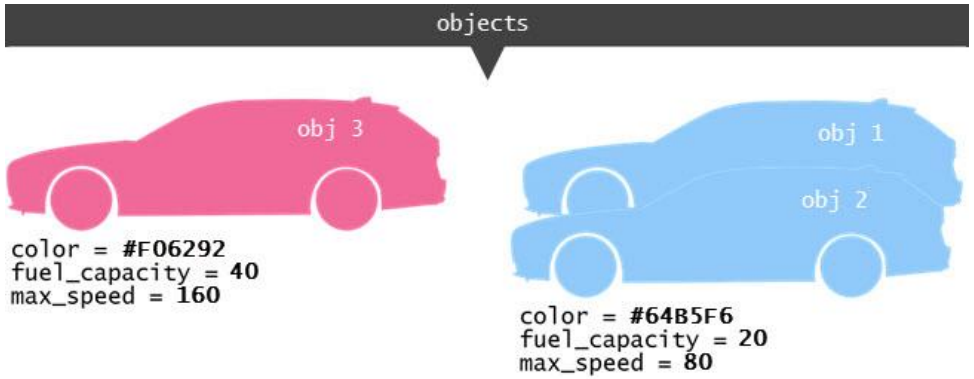
```
Colors c1=Colors.Red;
```

مفهوم کلاس در شی گرایی

در شی گرایی از کلاس برای تعریف ساختار - ویژگی ها و رفتار اشیا استفاده میشود

- Attributes
- Methods



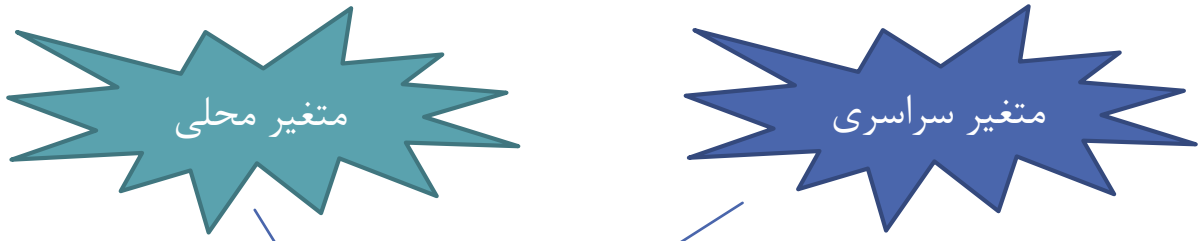


ساختار کلاس

```
public class Stock
{
    public string Name;
    public int ClosingPrice;
    public int LastPrice;


    public double PriceChange()
    {
        کد مربوط به محاسبه تغییر قیمت
    }
}
```

```
Stock Zob = new Stock();

Zob.Name = "ذوب";
Zob.ClosingPrice = 1500;
Zob.LastPrice = 1540;

double p = Zob.PriceChange();
```

محدوده دید متغیرها

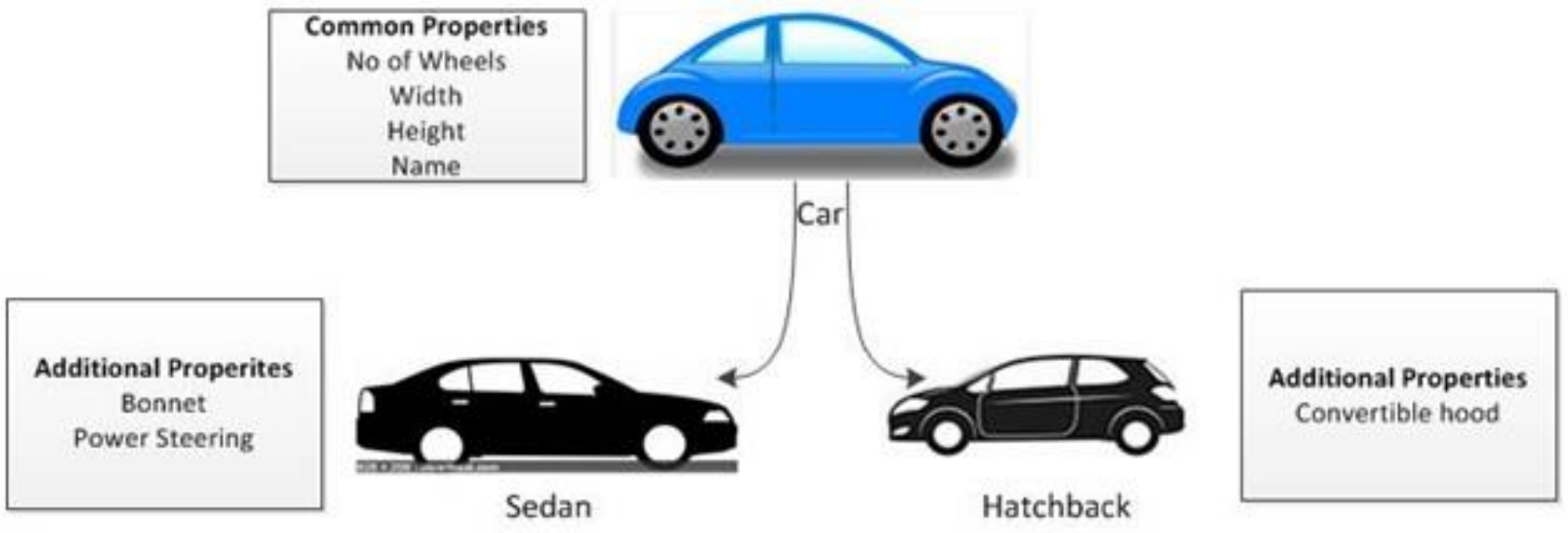



```
public class Stock
{
    int Pclosing;

    1 reference
    public void SetPrice(int newPrice)
    {
        int Pclosing=newPrice;
    }

    1 reference
    public int GetPrice()
    {
        return Pclosing;
    }
}
```

```
public class Stock
{
    int Pclosing;

    1 reference
    public void SetPrice(int newPrice)
    {
        Pclosing=newPrice;
    }

    1 reference
    public int GetPrice()
    {
        return Pclosing;
    }
}
```

```
Stock myStock=new Stock();

myStock.SetPrice(1500);

int lastPrice = myStock.GetPrice();
```

وراثت


Common Properties
No of Wheels
Width
Height
Name
Car
Additional Properites
Bonnet
Power Steering
Sedan
Hatchback
Additional Properties
Convertible hood

وراثت

```
public class Stock
{
    public string Name;
    public int ClosingPrice;
    public int LastPrice;


    public double PriceChange()
    {
        کد مربوط به محاسبه تغییر قیمت
    }
}
```

```
public class ETF : Stock
{
    public int NAV;
}
```

Overriding

```
public class Stock
{
    public string Name;
    public int ClosingPrice;
    public int LastPrice;


    public virtual double PriceChange()
    {
        کد مربوط به محاسبه تغییر قیمت
    }
}
```

```
public class ETF : Stock
{
    public int NAV;


    public override double PriceChange()
    {
        کد مربوط به محاسبه تغییر قیمت
    }
}
```

نحوه نوشتن کد در سامانه
QuantoRythm

نمای بازار


سهامداران
EPS
DPS
آمارها
معرفی
ترازنامه
سهامي ذوب آهن اصفهان (ذوب) - بازار دوم فرابورس
خرید 1,879
معامله 1,881 47 (2.56%)
فروش 1,880
بازه روز 1,909
قیمت مجاز 1,925
بازه هفته 1,935
بازه سال 3,981
اولین 1,830
پایانی 1,879 45 (2.45%)
دیروز 1,834
تعداد معاملات 8,236
حجم معاملات 117.041 M
ارزش معاملات 219.872 B
ارزش بازار 115,565.149 B
تعداد سهام
حجم مبنا
سهام شناور
میانگین حجم ماه
آخرین معامله 12:29:55
وضعیت مجاز
EPS 77
P/E 24.4
تعداد حجم خرید فروش تعداد حجم
3 28,939 1,879 1,880 2 55,000
3 105,375 1,878 1,881 1 17,574
6 107,200 1,877 1,882 2 77,400

کلاس الگوریتم

```
public class MyAlgorithm : AlgorithmBase
{
    // متغییر هایی که در این قسمت تعریف شوند در طول زمان اجرای الگوریتم
    // قابل دسترسی هستند

    2 references
    protected override object DecidePrimitive(AlgorithmInputEvents inputType, out Exception exception)
    {
        // کدهایی که در این قسمت نوشته میشوند در هر ثانیه یکبار اجرا میشوند

    }

    1 reference
    protected override void Init()
    {
        // کدهایی که در این قسمت نوشته میشوند فقط یکبار در زمان شروع الگوریتم اجرا میشوند

    }

}
```

IcoSMA(“IRO3ZOBZ0001”,TimeFrameEnum.D1,50,AppliedPriceEnum.Close)

نحوه دریافت اطلاعات بازار و معاملات

```
var omsData = GetOmsData("123_12");

var marketdata = GetMarketData("IRO3ZOBZ0001");


var hasPendingOrder = omsData.HasPendingOrderFor("IRO3ZOBZ0001");
var hasOpenorder = omsData.HasOpenOrderFor("IRO3ZOBZ0001");
var totalExecuted = omsData.SumExecutedTradeFor("IRO3ZOBZ0001");
```

نمونه الگوریتم

برای سهم ذوب آهن:

- درصد تغییراخرین معامله پایینتر از **-2** درصد رسید سفارش خرید
- تغییرات اخرین معامله بیشتر از **+3** درصد رسید سفارش فروش

- قیمت سفارش به مبلغ اخرین معامله باشد
- حجم ۲۵۰۰

```
protected virtual object DecidePrimitive(AlgorithmInputEvents inputType, out Exception exception)
{
    var omsData = GetOmsData("123_12");
    var myAccount = GetAccount("123_12");
    var marketdata = GetMarketData("IRO3ZOBZ0001");


    var hasPendingOrder = omsData.HasPendingOrderFor("IRO3ZOBZ0001");
    var hasOpenorder = omsData.HasOpenOrderFor("IRO3ZOBZ0001");
    var totalExecuted = omsData.SumExecutedTradeFor("IRO3ZOBZ0001");


    if (!hasOpenorder && !hasPendingOrder && totalExecuted < 2500 &&
        (marketdata.PLastChangePercent <= -2))
    {
        this.SendMyOrderNow(marketdata.PdrCotVal, 2500, "IRO3ZOBZ0001", myAccount, OrderSide.Buy);
    }


    if (!hasOpenorder && !hasPendingOrder && totalExecuted < 2500 && (marketdata.PLastChangePercent >= 3))
    {
        this.SendMyOrderNow(marketdata.PdrCotVal, 2500, "IRO3ZOBZ0001", myAccount, OrderSide.Sell);
    }
}
```

نمونه الگوریتم

برای سهم ذوب آهن:

- هر گاه برای سهم فنوال صف خرید تشکیل شود
- تغییرات اخرین معامله برای ذوب بیشتر از 3+ درصد باشد

به اندازه کل دارایی سهم ذوب به قیمت بهترین مظنه خرید انجام شود

```
protected virtual object DecidePrimitive(AlgorithmInputEvents inputType, out Exception exception)
{
    var omsData = GetOmsData("123_12");
    var myAccount = GetAccount("123_12");
    var marketdata = GetMarketData("IRO3ZOBZ0001");

    var fenovalMarketData = GetMarketData("IRO1NALM0001");
    bool isFenovalInBuyQueue = (fenovalMarketData.BestSellers.Qoutes.Count == 0) &&
                               (fenovalMarketData.BestBuyers.Qoutes.Count > 0);

    var hasPendingOrder = omsData.HasPendingOrderFor("IRO3ZOBZ0001");
    var hasOpenorder = omsData.HasOpenOrderFor("IRO3ZOBZ0001");
    var totalExecuted = omsData.SumExecutedTradeFor("IRO3ZOBZ0001");



    var buyerPrice = marketdata.BestBuyers.Qoutes[0].Price;
    int qty = (int)Math.Round((double)omsData.CreditInfo.Withdrawable / buyerPrice);

    if (!hasOpenorder && !hasPendingOrder && totalExecuted >0 &&
       isFenovalInBuyQueue &&
        (marketdata.PLastChangePercent >=3 ))
    {
        this.SendMyOrderNow(buyerPrice, qty, "IRO3ZOBZ0001", myAccount, OrderSide.Buy);
    }
}
```

نمونه الگوریتم

برای سهم ذوب آهن:

فرض کنید شما حجم زیادی از یک سهم به تعداد 50,000,000 را دارید و قصد فروش انرا دارید.

با توجه به وضعیت بازار میدانید که اگر تمام حجم را یکجا برای فروش وارد هسته معاملات کنید، بازار واکنش نشان داده و سهم شروع به پایین امدن میکند و شاید مجبور شوید به پایین ترین قیمت مجبور به معامله شوید.

برای جلوگیری از واکنش بازار تصمیم گرفته اید به جای یک سفارش با حجم بالا، چندین سفارش با حجم های کمتر و با فاصله زمانی 100ثانیه ثبت کنید.

هر سفارش به حجم 500,000 به قیمت بهترین مظنه خرید ، در صورتی که از سفارش قبلی 100 ثانیه گذشته باشد.

```
private DateTime _lastOrderTime = DateTime.MinValue;

0 references
protected object DecidePrimitive(AlgorithmInputEvents inputType, out Exception exception)
{

    var omsData = GetOmsData("123_12");
    var myAccount = GetAccount("123_12");
    var marketdata = GetMarketData("IRO3ZOBZ0001");

    var hasPendingOrder = omsData.HasPendingOrderFor("IRO3ZOBZ0001");


    var bestBuyerPrice = marketdata.BestBuyers.Qoutes[0].Price;
    var diffTime = (DateTime.Now - _lastOrderTime).TotalSeconds;

    if (!hasPendingOrder && diffTime > 100)
    {
        this.SendMyOrderNow(bestBuyerPrice, 500000, "IRO3ZOBZ0001", myAccount, OrderSide.Sell);
        _lastOrderTime = DateTime.Now;
    }

    exception = null;
    return null;
}
```

نمونه الگوریتم

برای سهم ذوب آهن:

- قیمت پایانی روز قبل بیشتر از ۱۱۸۰ باشد
- حجم معاملات انجام شده بیش از 1,500,000 باشد
- حجم سفارشات در صف اول و دوم صف فروش بیش از 320,000,000 باشد
- اندیکاتور Simple Macd بیست روزه عدد بالای ۱۸۰۰ را نشان دهد

- سفارش خرید به ارزش ۶۰ درصد مانده حسابداری ثبت شود که در مکان دوم صف خریداران قرار بگیرد.

```
protected virtual object DecidePrimitive(AlgorithmInputEvents inputType, out Exception exception)
{

 var omsData = GetOmsData("123_12");


 var myAccount= GetAccount("123_12");
 var marketdata = GetMarketData("IRO3ZOBZ0001");

 var change= marketdata.PClosing / marketdata.PriceYesterday;
 var hasPendingOrder= omsData.HasPendingOrderFor("IRO3ZOBZ0001");
 var hasOpenorder = omsData.HasOpenOrderFor("IRO3ZOBZ0001");
 var totalExecuted = omsData.SumExecutedTradeFor("IRO3ZOBZ0001");

 var buyerPrice = marketdata.BestBuyers.Qoutes[0].Price;
 var myBuyPrice = buyerPrice - 1;

 var qtyForFirstAndSecondRow = marketdata.BestSellers.Qoutes[0].Quantity + marketdata.BestSellers.Qoutes[1].Quantity;
 var totalTrades = marketdata.TradesVol;

 var sma = IcoSMA("IRO3ZOBZ0001", TimeFrameEnum.D1, 20, AppliedPriceEnum.Close);
 var amount =Math.Round(omsData.CreditInfo.Withdrawable * 0.8) ;
 int qty =(int)Math.Round( amount / myBuyPrice);

 if ( !hasOpenorder && !hasPendingOrder  && totalExecuted<2500 &&
     marketdata.PClosing >1180 &&
     totalTrades>1500000000 &&
     qtyForFirstAndSecondRow>320000000 &&
     sma >=1800
       )
 {
     this.SendMyOrderNow(myBuyPrice, qty, "IRO3ZOBZ0001", myAccount, OrderSide.Sell);
 }

}
```


Q
&
A