بسم اللّه الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

بررسی اجمالی اعداد

مدرس : مهندس افشین رفوآ

آدرس آموزشگاه : تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 561 - واحد 7
88146323 - 88446780 - 88146330

بررسی اجمالی اعداد

مقدمه

زبان C#، برخلاف زبان های C، C++، Pascal، Visual Basic و Java، از ریاضی پشتیبانی نمی کند (امکان پشتیبانی توکار برای ریاضی ندارد) و مجبور است این قابلیت را از کتابخانه ها یا زبان های دیگر وام بگیرد.

برای انجام ساده ترین عملیات جبری و هندسی در C#، می توانید از متدهای کلاس Math.NET Framework استفاده کنید. همچنین می توان از کتابخانه ی بسیار غنی توابع visual basic برای این منظور کمک گرفت. کتابخانه ی بالا یکی از گسترده ترین مجموعه توابع حوضه های مختلف ریاضی تجاری را در بر دارد.

راه های مختلفی برای تعریف متغیر (از) نوع عددی وجود دارد که نمونه های آن را زیر مشاهده می کنید.

```
using System;
class Program
{
            static int Main()
            {
                short   sNumber;
                int     iNumber;
                double  dNumber;
                decimal mNumber;
                return 0;
            }
        }
```

علامت اعداد

یکی از قوانین پایه و اولیه ی C# این است که پس از تعریف متغیر و پیش از به کار بردن آن، متغیر باید مقداردهی اولیه شده باشد. نمونه های مقداردهی اولیه متغیر در مثال به نمایش گذاشته شده است.

```
using System;
class Program
{
            static int Main()
            {
                short   sNumber = 225;
                int     iNumber = -847779;
```

```
                double  dNumber = 9710.275D;
                decimal mNumber = 35292742.884295M;
                Console.WriteLine("Short Integer:      {0}",
sNumber);
                Console.WriteLine("Integral Number:    {0}",
iNumber);
                Console.WriteLine("Double-Precision:   {0}",
dNumber);
                Console.WriteLine("Extended Precision: {0}",
mNumber);
                return 0;
            }
        }
```

نتیجه ی زیر را به دست می دهد.

```
Short Integer:      225
        Integral Number:    -847779
        Double-Precision:   9710.275
        Extended Precision: 35292742.884295
        Press any key to continue...
```

هنگام مقداردهی اولیه متغیر با یک ثابت (constant)، در واقع مشخص می کنید که متغیر نام برده مثبت باشد یا منفی و یا 0 که سرانجام علامت آن متغیر محسوب می شود. در صورتی که مقدار متغیر را از راه دیگری به دست آورید، ممکن است علامت آن مشخص نشود. اگر چه می توان با استفاده از عملگر های مقایسه علامت متغیر (مثبت یا منفی بودن عدد) را به دست آورد، اما کلاس math متدی در اختیار شما قرار می دهد که این کار را برای شما انجام می دهد.

برای این منظور باید متد Math.Sign() را فراخواند. البته متد مذکور در نسخه های مختلف که دستور نحوی (syntax) آن ها به شکل زیر است اضافه بارگذاری شده.

```
public static int Sign(sbyte   value);
public static int Sign(short   value);
public static int Sign(int     value);
public static int Sign(long    value);
public static int Sign(sbyte   value);
public static int Sign(double  value);
public static int Sign(decimal value);
```

هنگام فراخوانی این متد، متغیر یا مقدار دلخواه را به عنوان آرگومان ارسال کنید. متد نتایج زیر را بر می گرداند.

-1 چنانچه آرگومان منفی بود.

0 چنانچه آرگومان 0 بود.

1 چنانچه آرگومان مثبت بود.

مثال های فراخوانی متد

```
using System;
class Program
{
            static int Main()
            {
                short   sNumber = 225;
                int     iNumber = -847779;
                double  dNumber = 9710.275D;
                decimal mNumber = 35292742.884295M;
                Console.WriteLine("Number: {0} => Sign: {1}",
                                  sNumber, Math.Sign(sNumber));
                Console.WriteLine("Number: {0} => Sign: {1}",
                                  iNumber, Math.Sign(iNumber));
                Console.WriteLine("Number: {0} => Sign: {1}",
                                  dNumber, Math.Sign(dNumber));
                Console.WriteLine("Number: {0} => Sign: {1}\n",
                                  mNumber, Math.Sign(mNumber));
                return 0;
            }
        }
```

نتیجه

```
Number: 225 => Sign: 1
        Number: -847779 => Sign: -1
        Number: 9710.275 => Sign: 1
        Number: 35292742.884295 => Sign: 1
        Press any key to continue...
```

بخش صحیح عدد ممیز شناور

همان طور که در درس 3 تشریح کردیم، عدد ممیز شناور از دو بخش اصلی تشکیل شده، یک بخش عدد صحیح و یک بخش اعشار و این دو بخش مذکور توسط نقطه ی اعشار " . " از هم جدا می شوند. در برخی موارد

(عملیات) لازم است بخش عدد صحیح یک مقدار را بدست آورید. کلاس Math در این زمینه کمک شایانی به شما می کند.

برای به دست آوردن بخش عدد صحیح یک رقم دهدهی، کلاس Math با متد Trancate() این امکان را برای شما فراهم می کند. متد بالا در دو نسخه ی زیر اضافه بار گذاری شده و دستور نحوی آن به شرح زیر است.

```
public static double Truncate(double d);
public static double Truncate(double d);
```

هنگام فراخوانی این متد، یک عدد یا متغیر از نوع شناور، دابل و دهدهی به آن ارسال کنید. متد نام برده بخش عدد صحیح یک مقدار را باز می گرداند.

مثال

```
using System;
class Program
{
            static int Main()
            {
                float number = 225.75f;

                Console.WriteLine("The integral part of {0} is
{1}\n",
                                        number,
Math.Truncate(number));
                return 0;
            }
        }
```

نتیجه ی زیر به دست می آید.

```
The integral part of 225.75 is 225

        Press any key to continue...
```

کمینه ی دو مقدار

می توان کمینه ی دو مقدار را بدون نیاز به نوشتن کد (خود) بدست آورد. برای این منظور کلاس Math مجهز به متدی است به نام Min. متد مزبور در ورژن های متفاوت اضافه بارگذاری شده که هر نسخه با نوع داده ی integral یا floating-point تطبیق داده شده است. دستورهای نحوی آن ها به شرح زیر است.

```
public static byte    Min(byte    val1، byte    val2);
public static sbyte   Min(sbyte   val1، sbyte   val2);
public static short   Min(short   val1، short   val2);
public static ushort  Min(ushort  val1، ushort  val2);
public static int     Min(int     val1، int     val2);
public static uint    Min(uint    val1، uint    val2);
public static float   Min(float   val1، float   val2);
public static long    Min(long    val1، long    val2);
public static ulong   Min(ulong   val1، ulong   val2);
public static double  Min(double  val1، double  val2);
public static decimal Min(decimal val1، decimal val2);
```

مثال

```
using System;
class Program
{
            static int Main()
            {
                int number1 = 8025;
                int number2 = 73;
                Console.WriteLine("The minimum of {0} and {1} is
{2}"،
                    number1، number2، Math.Min(number1،
number2));
                return 0;
            }
        }
```

نتیجه ی زیر حاصل می گردد.

```
The minimum of 8025 and 73 is 73
        Press any key to continue...
```

به خاطر داشته باشید که می توان از کلیدواژه های var و dynamic برای تعریف متغیرهای مورد نظر استفاده کرد :

```
using System;
         class Program
```

```
{
            static int Main()
            {
                var     number1 = 8025;
                dynamic number2 = 73;
                Console.WriteLine("The minimum of {0} and {1} is
{2}",
                    number1, number2, Math.Min(number1,
number2));
                return 0;
            }
        }
```

بیشینه ی مقدار integer یک سری

برای بدست آوردن بیشینه ی دو عدد، می توان متد Max() را (از کلاس Math) فراخواند. دستور نحوی این متد به ترتیب زیر است.

```
public static byte    Max(byte    val1, byte    val2);
public static sbyte   Max(sbyte   val1, sbyte   val2);
public static short   Max(short   val1, short   val2);
public static ushort  Max(ushort  val1, ushort  val2);
public static int     Max(int     val1, int     val2);
public static uint    Max(uint    val1, uint    val2);
public static float   Max(float   val1, float   val2);
public static long    Max(long    val1, long    val2);
public static ulong   Max(ulong   val1, ulong   val2);
public static double  Max(double  val1, double  val2);
public static decimal Max(decimal val1, decimal val2);
```

مثال فراخوانی متد بالا

```
using System;
class Program
{
            static int Main()
            {
                int number1 = 8025;
                int number2 = 73;
                Console.WriteLine("The maximum of {0} and {1} is
{2}",
                    number1, number2, Math.Max(number1,
number2));
                return 0;
```

```
    }
}
```

نتیجه

```
The maximum of 8025 and 73 is 8025
        Press any key to continue...
```