دریافت اعداد

با توجه به این که داده های دریافتی به وسیله متد ReadLine() همواره به صورت رشته تحویل داده می شود باید به وسیله دستوری، رشته دریافتی را به عدد تبدیل کنیم .بنابراین به متدی نیاز داریم که بتواند یک رشته شامل ارقام را به ارزش عددی تبدیل کند تا بتوانیم روی آنها محاسبات ریاضی انجام دهیم. خوشبختانه برای انواع داده های عددی، متدی به نام Parse() از قبل تعریف شده است که می تواند از یک رشته شامل ارقام، معادل عددی آن را بدست آورد.

مثال۳: می خواهیم برنامه ای بنویسیم که دو عدد دلخواه از کاربر دریافت کند و مجموع آن ها را حساب کرده و روی صفحه نمایش، نشان دهد.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;


namespace sum
    class Program
    {
        static void Main(string[] args)
        {

             string firstNumber, secondNumber;

             Console.Write("Enter a number:" );
             firstNumber = Console.ReadLine();

             Console.Write("Enter another number:" );
             secondNumber = Console.ReadLine();

             Console.WriteLine("Total="+ (firstNumber + secondNumber) );

             Console.ReadKey();


        }
    }
}
```

نکته: با اجرای این برنامه، پنجره ای ظاهر می شود که از کاربر خواسته می شود که یک عدد وارد کند. پس از وارد کردن یک عدد و زدن دکمه Enter عدد دیگری خواسته می شود .فرض کنید اعداد ۱۳ و ۷۷ توسط کاربر وارد شود.

کاربر با وارد کردن این دو عدد انتظار داشت که مجموع آن دو یعنی ۹۰ را مشاهده کند اما به جای آن عدد ۱۳۷۷ نمایش داده شد.

متد ReadLine() داده دریافتی را به صورت یک رشته در حافظه ذخیره می کند، بنابراین علامت + در دستور عمل الحاق دو رشته مثلاً" 13 " و " 77 "را انجام می دهد و طبیعی است که نباید انتظار عمل جمع ریاضی داشته باشیم.
بنابراین برنامه را به صورت زیر تکمیل می کنیم:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;


namespace sum
    class Program
    {
        static void Main(string[] args)
        {
             string input;
             float firstNumber, secondNumber;

             Console.Write("Enter a number:" );
             input = Console.ReadLine();
             firstNumber = float.Parse(input);

             Console.Write("Enter another number:" );
             input = Console.ReadLine();
             secondNumber = float.Parse(input);

             Console.WriteLine("Total="+ (firstNumber +
secondNumber) );

             Console.ReadKey();


        }
    }
}
```

دستورهای شرطی

در بیشتر برنامه های کاربردی، لازم است مقدار داده ها را بررسی و مقایسه کنیم .سپس بر اساس نتیجه حاصل از بررسی، دستور یا دستورهایی را اجرا کنیم .به عبارت دیگر، در برنامه ها لازم است بتوانیم بر اساس مقدار داده ها، تصمیم گیری کنیم .برنامه هایی که تاکنون نوشته ایم تمام دستورهای داخل متد Main()پشت سرهم و به نوبت اجرا می شدند .اکنون با دستورهای شرطی ، آشنا می شویم که به وسیله آنها اجرای دستورها و پردازش آنها، کنترل می شوند و رفتار برنامه بر اساس وضعیت داده ها تغییر می کند.

- ### دستور شرطی if

در زبان برنامه نویسیC# از دستور if (با حروف کوچک نوشته می شود) برای کنترل اجرای دستورها و بررسی شرط، استفاده می شود. ساختار کلی دستور if به صورت زیر است:

```
if (عبارت منطقی)
دستور ;
```

توجه داشته باشید که دستور شرطی if از سه بخش تشکیل شده است:
کلمه رزرو شده if، عبارت منطقی داخل پرانتز و دستوری که در صورت درست (true) بودن نتیجه عبارت، اجرا خواهد شد.

- ### دستور شرطی if-else

شکل کلی این دستور به صورت زیر است:

```
if (عبارت منطقی)
دستور شماره ۱ ;
else
دستور شماره ۲ ;
```

مثال۴: برنامه ای بنویسید که عدد صحیحی را از کاربر دریافت کند، زوج یا فرد بودن عدد را تشخیص دهد.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace EvenOdd
{
    class Program
    {
        static void Main(string[] args)
        {
             string input;
             int number;

             Console.Write("Enter a number:" );
             input = Console.ReadLine();
             number = int.Parse(input);

            if((number%2==0))
             Console.WriteLine("The number is Even" );
            else
                Console.WriteLine("The number is Odd");
             Console.ReadKey();


        }
    }
}
```

مثال ۵: می خواهیم برنامه ای بنویسیم که عدد بزرگتر را از بین دو عدد دریافتی از کابر نمایش دهد.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace FindMaximum
{
    class Program
    {
        static void Main(string[] args)
        {
            string input;
            int firstNumber , secondNumber ;
             Console.Write("Enter a number:");
             input = Console.ReadLine();
             firstNumber = int.Parse(input);

             Console.Write("Enter another number:");
             input = Console.ReadLine();
             secondNumber = int.Parse(input);

             int biggerNumber = firstNumber;
             if (secondNumber>biggerNumber)
             biggerNumber = secondNumber;
             Console.WriteLine("The maximum number is:" + biggerNumber);

            Console.ReadKey();
        }
    }
}
```

- **دستور Switch**

در مواردی که بخواهیم حالت های مختلف یک عبارت را بررسی و بر اساس آن دستورهایی را اجرا کنیم از دستور switch استفاده می کنیم .ساختار کلی این دستور را می بینید:

```
switch (عبارت)
{
case مقدار ۱:
دستور۱ ;
break;

case مقدار ۲ :
دستور۲ ;
break;
...
default:
دستورهای دیگر;
break;
}
```

دستورهای تکرار (حلقه ها)

در بعضی از برنامه های کاربردی باید یک عمل چندین بار تکرار شود .مثلاً اگر بخواهیم میانگین یا معدل نمرات درس زبان انگلیسی یک کلاس را محاسبه کنیم، باید نمرات تمام دانش آموزان کلاس را از ورودی دریافت کرده و با یکدیگر جمع کنیم .در این مثال، عمل دریافت نمره از ورودی و عمل جمع زدن نمره ها، به تعداد دانش آموزان کلاس، باید تکرار گردد .نوشتن چنین برنامه هایی با دستورات تکراری، خسته کننده و طولانی و گاهی غیر ممکن خواهد بود .

در زبان های برنامه نویسی از جمله زبان C#دستورات ایجاد حلقه، برای کوتاه کردن تعداد دستورات برنامه، پیش بینی شده اند .به وسیله این دستورات، برنامه نویس می تواند، عملیات و پردازش های تکرار شونده را فقط یک بار بنویسد و کامپیوتر آنها را به دفعات، تکرار کند .

- ## دستورهای تکرار شرطی

۱. دستور حلقه شرطی while: ساختار کلی در زیر نشان داده شده است:

```
while (عبارت منطقی)
دستور;
```

۲. دستور حلقه شرطی do-while: ساختار کلی در زیر نشان داده شده است:

```
do
دستور;
while (عبارت منطقی) ;
```

مثال ۶: می خواهیم برنامه ای بنویسیم که ارقام یک عدد را جدا نموده و آنها را نمایش دهد.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Numbers
{
    class Program
    {
        static void Main(string[] args)
        {
            int number, digit ;
            string input;
             Console.Write("Enter a number:");
             input = Console.ReadLine();
             number = int.Parse(input);
             while (number > 0)
                {
                   digit = number % 10;
                   Console.WriteLine(digit);
                   number /= 10;
                 }

              Console.ReadKey();
    }
}
}
```

مثال ۷: می خواهیم برنامه ای بنویسیم که نام کاربری و رمز عبور را سؤال نماید و اگر کاربر اطلاعات خواسته شده را به درستی وارد نکرد، دوباره سؤال شود.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Login
{
    class Program
    {
        static void Main(string[] args)
        {
            string userName, password;
             bool loginFlag;
             do
                  {
                     Console.Write("Enter username: ");
                     userName = Console.ReadLine();
                     Console.Write("Enter password: ");
                     password = Console.ReadLine();
                     if ((userName == "admin") && (password ==
"admin123"))
                           loginFlag = true;
                       else
                           {
                             loginFlag = false;
                             Console.WriteLine("Wrong username or
password!. Try again.");
                           }
                   }
             while (!loginFlag);

            Console.WriteLine("Welcome Admin.");

              Console.ReadKey();
    }
}
}
```

تمرین: می خواهیم یک بازی حدس عدد، ایجادکنیم .این بازی بین دو بازیکن به شرح زیر صورت می گیرد: بازیکن اول عددی را برای خود در نظر می گیرد و بازیکن دوم باید آن عدد را حدس بزند (بازیکن اول در طول بازی، راهنمایی لازم را در اختیار بازیکن دوم قرار می دهد تا عدد بالاتر یا پایین تری را حدس بزند).