زنگ سی‌شارپ – قسمت نهم

نوشته‌ی مسعود درویشیان

لینک مستقیم این مطلب در وب‌تارگت

در قسمت قبل با دو عمل‌گر Increment و Decrement و چند عمل‌گر ریاضی دیگر آشنا شدیم که دانستن آن‌ها برای کار با حلقه‌ها و بسیاری از موارد دیگر ضروری است.

حلقه‌ی for (The for loop)

در برنامه‌نویسی همیشه مواردی پیش می‌آید که نیاز است یک بخش از کد چندین مرتبه اجرا و یک کار به‌صورت مکرر چندین مرتبه انجام شود. برای این منظور از باید حلقه‌ها استفاده کرد. یکی از این حلقه‌ها که استفاده‌ی زیادی در برنامه‌نویسی دارد حلقه‌ی for است. به‌عنوان مثال فرض کنید می‌خواهید برنامه‌ای بنویسید که ۵ مرتبه پیغام خوش‌آمدگویی را چاپ کند. آیا برای این‌کار پنج مرتبه این پیغام را به‌صورت دستی می‌نویسید؟ مسلماً این‌کار زمان‌بر است و اگر قصد داشته باشید ۱۰۰ مرتبه پیغام خوش‌آمدگویی را به‌صورت دستی تایپ کنید، مدت زمان زیادی از وقت شما صرف می‌شود. در این‌جا حلقه‌ی for به شما کمک می‌کند تا از خط‌کد تکراری بپرهیزید. البته از این حلقه به منظورهای دیگر هم استفاده می‌شود که در آینده با آن‌ها آشنا خواهیم شد.

حلقه‌ی for مانند دستورات دیگر شکل و فرم خاص خودش را دارد که با زبان‌های C, C++ و جاوا متشابه است. شکل و فرم کلی حلقه‌ی for را می‌توانید در زیر ببینید:

```
for(initialization; condition; iteration)
{
   statement sequence
}
```

در قسمت initialization (مقدار دهی اولیه) معمولاً متغیری قرار داده می‌شود که این متغیر کنترل‌کننده حلقه است و به‌عنوان شمارنده (counter) حلقه عمل می‌کند. در قسمت condition یک عبارت بولین (true یا false) قرار می‌گیرد که مشخص می‌کند حلقه به چه تعداد باید تکرار شود. قسمت iteration مقدار متغیر کنترل‌کننده (متغیری که در قسمت initialization قرار دارد) را در هربار که حلقه تکرار می‌شود به‌روز رسانی می‌کند و تغییراتی را روی آن اعمال می‌کند. توجه داشته باشید که این سه قسمت (initialization و condition و iteration) باید حتماً توسط

سمی‌کالن از هم جدا شوند. حلقه‌ی for تا زمانی‌که مقدار condition برابر با true است تکرار شده و به‌محض این‌که condition برابر با false شد برنامه از حلقه خارج می‌شود.

به مثال زیر توجه کنید:

```
using System;
class Example
{
    static void Main()
    {
        int i;

        for (i = 0; i < 5; i++)
        {
            Console.WriteLine(i);
        }

        Console.WriteLine("Done!");
    }
}
```

خروجی:

```
0

1

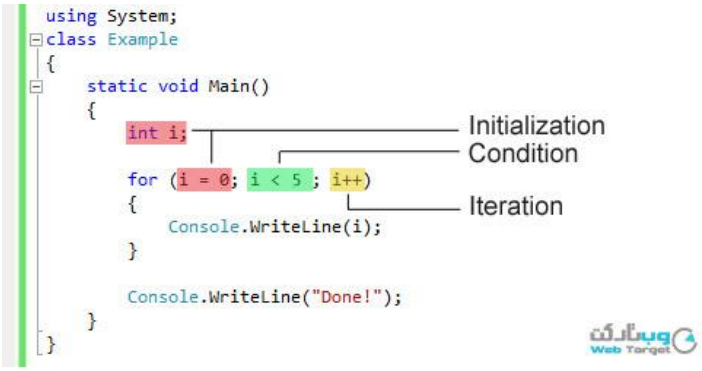
2

3

4

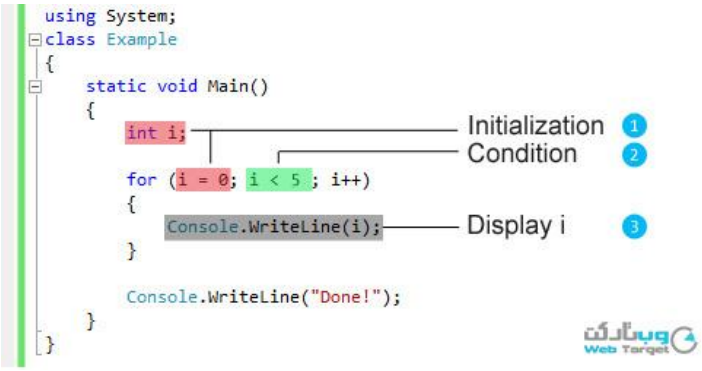
Done!
```

اکنون این برنامه را به‌طور کامل بررسی می‌کنیم تا بدانیم دقیقاً چه اتفاقی در حال رخ دادن است. هنگامی که کامپایلر شروع به خواندن کدها می‌کند، ابتدا متغیر i تعریف می‌شود سپس کامپایلر با یک حلقه‌ی for برخورد می‌کند.

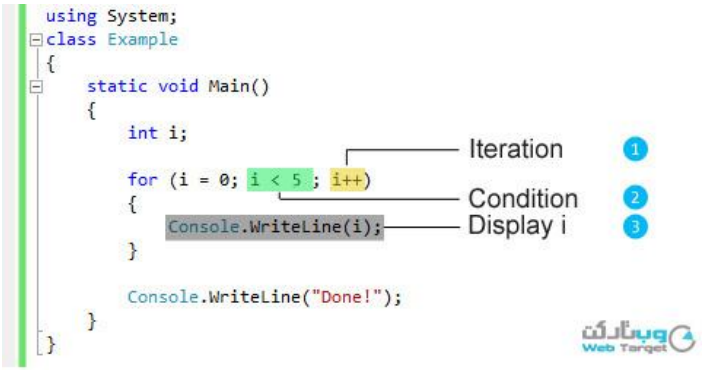


بر روی این حلقه به متغیر i مقدار صفر اختصاص داده می‌شود (initialization) سپس مقدار i با عدد ۵ مقایسه می‌شود تا مشخص شود که آیا مقدار i از عدد ۵ کوچک‌تر است یا خیر (condition). اگر i کوچک‌تر بود پس condition برابر با true است، اگر کوچک‌تر نبود condition برابر با false می‌شود. در حال حاضر مقدار i برابر با صفر است، صفر کوچک‌تر از ۵ است، بنابراین condition برابر با true می‌شود. حال‌که condition برابر با true شد کامپایلر وارد حلقه می‌شود و محتوای درون حلقه را اجرا می‌کند. در این‌جا دستور چاپ مقدار i قرار دارد، بنابراین مقدار کنونی i نمایش نمایش داده می‌شود (همان‌طور که می‌دانید مقدار کنونی i برابر با صفر است). **پس‌از این‌که** مقدار i نمایش داده شد، توسط عمل‌گر افزایشی پسوندی یک واحد به مقدار i افزوده می‌شود (iteration) و مجدداً مقدار i با عدد ۵ مقایسه می‌شود. مقدار i اکنون برابر با ۱ است و عدد ۱ از عدد ۵ کوچک‌تر است، پس شرط برقرار است و مجدداً کامپایلر وارد حلقه شده و مقدار جدید i را نمایش داده و **سپس** مقدار i را یک واحد افزایش می‌دهد. این روند همین‌طور ادامه دارد تا زمانی که مقدار i به ۵ برسد. در این لحظه مقدار i که برابر با ۵ است با عدد ۵ مقایسه شده و به‌دلیل این‌که ۵ از ۵ کوچک‌تر نیست condition برابر با false می‌شود و کامپایلر دیگر وارد حلقه نشده و از آن خارج می‌شود و به سراغ ادامه‌ی کدها می‌رود. در ادامه یک پیغام Done! نمایش داده شده و برنامه به اتمام می‌رسد.

روند اجرای این حلقه را در شکل زیر می‌بینید. ابتدا مقداردهی اولیه انجام شده و شرط بررسی می‌شود، سپس در صورت برقراری شرط، محتوای بلاک حلقه اجرا می‌شود:



سپس مقدار i یک واحد افزایش یافته و مجدداً شرط بررسی می‌شود و درصورت برقراری شرط، محتوای بلاک حلقه اجرا می‌شود:



به نمونه‌ی زیر دقت کنید:

```
using System;
class Example
{
    static void Main()
    {
        int count, x;
        x = 0;
        for (count = 10; count < 5; count++)
        {
            x += count; // this statement will not execute
        }
    }
}
```

در این برنامه به‌دلیل این‌که شرط حلقه برقرار نیست، هیچ‌گاه حلقه اجرا نمی‌شود.

توضیحات تکمیلی حلقه‌ی for در قسمت‌های بعدی مورد بحث قرار می‌گیرد.

حلقه‌ی while (The while Loop)

یکی دیگر از حلقه‌های سی‌شارپ، حلقه‌ی while است. فرم کلی حلقه‌ی while به‌شکل زیر است:

```
while (Condition)
{
   statement sequence
}
```

در قسمت condition شرط حلقه بررسی می‌شود که کنترل کننده‌ی حلقه است و می‌تواند هر نوع عبارت بولینی باشد. تا زمانی که مقدار condition برابر با true است قسمت statement اجرا می‌شود.

به مثال زیر که قبلاً آن‌را با استفاده از حلقه‌ی for انجام دادیم توجه کنید:

```
using System;
class Example
{
```

```
    static void Main()
    {
        int i = 0;
        while (i < 5)
        {
            Console.WriteLine(i);
            i++;
        }
        Console.WriteLine("Done!");
    }
}
```

اگر این برنامه را اجرا کنید متوجه خواهید شد که نتیجه‌ی یکسانی با مثال انجام‌شده‌ی حلقه‌ی for دارد. در این حلقه، قسمت iteration داخل حلقه قرار دارد و مقداردهی اولیه بیرون از حلقه قرار داده شده است. در این مثال تا زمانی‌که مقدار condition برابر با true است، حلقه اجرا می‌شود.

تمرین

تمرین شماره ۱: با استفاده از حلقه‌ی for برنامه‌ای بنویسید که اعداد زوج ۱ تا ۱۰۰ را چاپ کند.

تمرین شماره ۲: با استفاده از حلقه‌ی for برنامه‌ای بنویسید که اعداد ۱ تا ۱۰۰ را از انتها تا ابتدا چاپ کند.

تمرین شماره ۳: با استفاده از حلقه‌ی for برنامه‌ای بنویسید که اعداد فرد ۱ تا ۱۰۰ را چاپ کند.

تمرین شماره ۴: با استفاده از حلقه‌ی for برنامه‌ای بنویسید که مضارب ۵ اعداد ۱ تا ۱۰۰ را چاپ کند.

تمرین شماره ۵: تمرین‌های ۱ تا ۴ را با استفاده از حلقه‌ی while انجام دهید.

تمرین‌ها باید توسط شما خوانندگان عزیز انجام شود. در قسمت بعدی حل این تمرین‌ها با توضیحات در وب‌تارگت قرار داده خواهد شد ولیکن سعی داشته باشید ابتدا خودتان در حل تمرین‌ها فکر و تلاش لازم را انجام دهید.

کلیه حقوق مادی و معنوی برای وب‌سایت وب‌تارگت محفوظ است.
استفاده از این مطلب در سایر وب‌سایت‌ها و نشریات چاپی تنها با ذکر و درج لینک منبع مجاز است.