کارگاه کامپیوتر

جلسه : **ششم**

مدرس : **الهام یوسفی**

دستورات کنترلی

ساختار تکرار for و while

حلقه FOR

- این ساختار برای تکرار یک یا چند دستور ایجاد می‌شود.
- معمولاً در مواردی که تعداد تکرار حلقه مشخص باشد از دستور for استفاده می‌شود.
- در این ساختار از متغیری برای کنترل تعداد حلقه استفاده می‌شود که آن را **شمارنده** یا **اندیس حلقه تکرار** گویند.
- اندیس حلقه دارای یک مقدار اولیه است و در هر بار تکرار حلقه (اجرای دستورات حلقه) مقداری به آن اضافه می‌شود.
- مقداری را پس از هر بار اجرای دستورات حلقه به اندیس حلقه افزوده می‌شود **گام حرکت** گویند.
- گام حرکت می‌تواند عددی **صحیح** و **اعشاری** , **مثبت** (گام افزایشی) یا **منفی** (گام کاهشی) و **کاراکتری** باشد.
- درصورتیکه در قسمت شرط حلقه، شمارنده با یک متغیر که از ورودی دریافت شده است مقایسه شود یک **حلقه متغیر** داریم به عبارت دیگر تعداد تکرار دستورات حلقه، نامشخص می باشد.
- اگر در شرط حلقه، متغیر با مقدار ثابتی مقایسه شود، تعداد تکرار حلقه مشخص می‌باشد و به آن **حلقه ثابت** گویند.
- اگر در جلوی for علامت ; قرار داده شود، این دستور تا موقعی که شرط حلقه برقرار باشد اجرا می‌شود و پس از نقض شرط حلقه، از حلقه خارج شده و دستورات بعد از آن اجرا می‌شوند. (یعنی اگر زیر دستوری برای حلقه نوشته شده باشد به عنوان دستور حلقه اجرا نمی‌شود)
- در صورتیکه شرط حلقه وجود نداشته باشد ، یک حلقه بینهایت داریم. دستور for (;;) نیز حلقه بینهایت تولید می‌کند.
- حلقه بینهایت هیچ وقت متوقف نخواهد شد. در چنین مواقعی برای توقف اجرای برنامه از کلید های Break + Ctrl استفاده می‌شود.

ساختار تکرار حلقه FOR

- ساختار کلی دستور for :

```
for ( مقداردهی اولیه برای شمارنده (متغیر) حلقه ; شرط حلقه ; (تغییر مقدار شمارنده (گام حلقه
{
  دستورات;
}
```

- حلقه بینهایت:

```
for (;;)
{
  دستورات;
}
```

مثال

- با استفاده از حلقه for اعداد صحیح ۱ تا ۵، اعداد اعشاری ۱ تا ۵ با گام ۰.۵ و حروف a تا z را تولید کنید.

```
#include <iostream>
using namespace std;
main()
{
    int i;
    float j;
    char c;
    cout <<"Integer Number\n";
    for (i=1; i<6 ; i++)
        cout << i <<endl;

    cout <<"\n\nFloat Number\n";
    for (j=1; j<=5 ;j+=0.5)
        cout << j <<"\t";

    cout <<"\n\nCharacter\n";
    for (c='a'; c<='z' ; c++ )
        cout << c <<"\t";

    for (i=1; i<6 ; i++);
        cout << "\nBye" <<endl;
}
```

```
Integer Number
1
2
3
4
5


Float Number
1       1.5     2       2.5     3       3.5     4       4.5     5

Character
a       b       c       d       e       f       g       h       i       j       k       l       m       n       o
p       q       r       s       t       u       v       w       x       y       z
```

یک حلقه ثابت، چون تعداد تکرار حلقه مشخص است.

به دلیل اینکه انتهای حلقه ; گذاشته شده است تا زمانیکه شرط برقرار باشد به i یکی اضافه می‌کند و دستور خط بعد به عنوان زیر دستور آن به حساب نمی‌آید.

مثال (حلقه بینهایت)

- برنامه ای بنویسید که بینهایت بار عبارت Hello را چاپ نماید.

```
#include <iostream>
using namespace std;
main()
{
    int i=1;
    for (;;)    // for (; i<10 ;)
        cout << "Hello :) \t";
}
```

مثال (حلقه متغیر، گام کاهشی)

- برنامه ای بنویسید عدد n را از ورودی دریافت کند سپس اعداد n تا ۱ را در خروجی نمایش دهد. (تعداد تکرار حلقه متغیر است و بستگی به تعداد n دارد)

```
#include <iostream>
#include <conio.h>
using namespace std;
main()
{
    int i,n;
    cout <<"Enter n = ";
    cin >> n;
    for (i=n; i>=1 ; i--)
        cout <<i <<"\t";
    getch();
}
```

```
Enter n = 10
10      9       8       7       6       5       4       3       2       1
```

فعالیت

- برنامه ای بنویسید که عددی را از ورودی دریافت کند سپس مجموع اعداد زوج ۱ تا آن عدد را در خروجی چاپ کند.

```
#include <iostream>
#include <conio.h>
using namespace std;
main()
{
    int i,n,sum=0;
    cout <<"Enter n = ";
    cin >> n;
    for (i=1; i<=n ; i++)
        if (i%2==0)
            sum+=i;

    /*
    for (i=0; i<=n ; i+=2)
        sum+=i;*/

    cout <<"\nSum= "<<sum;
    getch();
}
```

```
Enter n = 5

Sum= 6
```

حلقه های تو در تو

- اگر حلقه‌ای داخل حلقه دیگری قرار بگیرد، اصطلاحاً حلقه های تودرتو گفته می‌شود.
- به ازای هر بار تکرار حلقه خارجی , حلقه داخلی به طور کامل اجرا می‌شود.

```
for ( مقداردهی اولیه برای شمارنده حلقه ; شرط حلقه ; تغییر مقدار شمارنده )
{
    دستورات;
    for ( مقداردهی اولیه برای شمارنده حلقه ; شرط حلقه ; تغییر مقدار شمارنده )
        دستورات;
}
```

مثال

- برنامه ای بنویسید که جدول ضرب n در n تولید کند. (n از کاربر دریافت می‌شود)

```
#include <iostream>
#include <conio.h>
using namespace std;
main()
{
    int i,j,n;
    cout <<"Enter n= ";
    cin >>n;
    for (i=1; i<=n ; i++)
    {
        for(j=1 ; j<=n ;j++)
            cout <<i*j <<"\t";
        cout <<endl;
    }

    getch();
}
```

```
Enter n= 5
1       2       3       4       5
2       4       6       8       10
3       6       9       12      15
4       8       12      16      20
5       10      15      20      25
```

حلقه WHILE

- این ساختار تا زمانیکه شرط برقرار باشد زیر دستورات را تکرار می‌کند.
- معمولاً برای برنامه هایی که در آن تعداد دفعات مشخص نباشد به کار می‌رود.
- برنامه هایی که دارای شمارنده می‌باشند، شمارنده در حلقه while باید توسط برنامه نویس کنترل شود در غیر اینصورت برنامه در حلقه بینهایت قرار می‌گیرد.

ساختار حلقه WHILE

- ساختار کلی دستور while :

```
while ( شرط حلقه)
{
  دستورات;
}
```

- حلقه بینهایت:

```
while (1)
{
  دستورات;
}
```

یا :

```
while (true)
{
  دستورات;
}
```

مثال

- برنامه ای بنویسید که اعداد کوچکترمساوی ۱۰ را چاپ نماید.

```
#include <iostream>
#include <conio.h>
using namespace std;
main()
{
    int i=1;
    while (i<=10)
    {
        cout <<i <<"\t";
        i++;
    }
    getch();
}
```

```
1    2    3    4    5    6    7    8    9    10
```

فعالیت

- با استفاده از حلقه while برنامه ای بنویسید عدد n را از ورودی دریافت کند سپس اعداد n تا ۱ (شامل ۱ هم باشد) را در خروجی نمایش دهد.

```
#include <iostream>
#include <conio.h>
using namespace std;
main()
{
    int n,i;
    cout <<"Enter n = ";
    cin >> n;
    while (n>=1)
    {
        cout << n<<endl;
        n--;
    }

    /*
    i=n;
    while (i>=1)
    {
        cout << i<<endl;
        i--;
    }
    */

    getch();
}
```

```
Enter n = 10
10
9
8
7
6
5
4
3
2
1
```

تمرین

۱- برنامه ای بنویسید دو عدد n و m را از ورودی دریافت کند سپس مجموع اعداد فرد بین دو عدد n و m را در خروجی نمایش دهد. (با فرض $n < m$)

۲- عددی را از ورودی دریافت کند فاکتوریل آن را چاپ کند. (با while)

مثال : $4! = 1 \times 2 \times 3 \times 4$

۳- برنامه ای بنویسید عددی دریافت کرده، مجموع ارقامش را چاپ نماید. (برای اینکه بتوان ارقام یک عدد را جدا کرد باید بصورت متوالی عدد را بر ۱۰ تقسیم نمود تا زمانی که خارج قسمت آن صفر شود.)

۴- برنامه ای بنویسید که تا بینهایت عدد از کاربر دریافت کند و نشان دهد عدد اول است یا اول نیست. (عددی اول است که فقط بر یک و خودش بخش پذیر باشد . برای دریافت بینهایت عدد از حلقه بینهایت استفاده کنید.)

راهنمایی : برای اینکه نشان دهیم عدد اول هست باید با استفاده از حلقه، اعداد ۲ تا عدد دریافتی از کاربر، تولید شود در صورتیکه عدد دریافتی بر اعداد تولید شده بخش پذیر باشد (مقسوم علیه) یعنی عدد اول نیست، اما اگر عدد دریافتی بر هیچکدام از اعداد تولیدی بخش پذیر نباشد یعنی عدد اول است.