ساختارهای کنترلی در برنامه نویسی C++ (Control Structures)

بطور کلی، در هر برنامه نویسی اجرای دستورات از اولین سطر شروع شده و به ترتیب تا آخرین سطر ادامه میابد.

اما گاهی وقتها لازم است که یک دستور چندین بار تکرار شود و یا اینکه تحت شرایط خاصی اجرا گردد و یا از اجرای آن جلوگیری شود.

ساختارهای کنترلی به برنامه نویس این اجازه را می دهند که بر روی دستورات کنترل داشته باشد و آنها را تکرار، اجرا و متوقف سازد. در این فصل به بررسی این ساختارها می پردازیم که به دو دسته تقسیم می شوند:

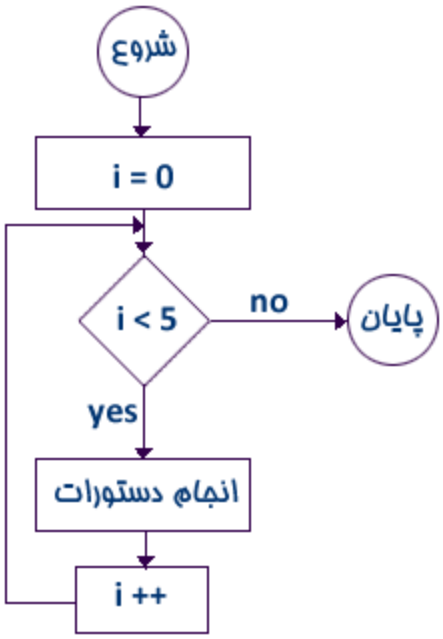
- ساختارهای کنترل
- ساختارهای تصمیم

ساختارهای کنترل:

- for
- while

در برنامه نویسی C++از ساختار حلقه for برای تکرار یک سری از دستورات استفاده می شود و شکل کلی ایجاد حلقه تکرار بصورت زیر است:

```
for( مقدار اولیه حلقه ; شرط حلقه ; گام حرکت )
{
   Line Command 1;
   Line Command 2;
   Line Command n;
}
```



مثال) با استفاده از حلقه تکرار for کدی بنویسید که اعداد ۱ تا ۱۰ را چاپ کند.

```
1  for ( int i = 1; i <= 10; i++ )
2  {
3     cout << i << " ";
4  }
```

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

در کد بالا متغیر i از نوع int تعریف شده و شرط خاتمه حلقه for تا زمانی است که i کوچکتر یا مساوی ۱۰ باشد. گام حرکت حلقه یک است و با هر بار تکرار حلقه یک واحد به متغیر اضافه خواهد شد. متغیر iدر پایان حلقه و با خروج از آن یک واحد اضافه خواهد شد. زمانیکه i برابر با ۱۱ می شود برنامه با رسیدن به شرط حلقه و عدم برقرار بودن شرط، از حلقه خارج می شود. برای چاپ اعداد باید در درون خود حلقه، دستور خروجی را نوشت.

در برنامه نویسی، برای تست دستورات روشی به نام Trace وجود دارد که در این مورد از دستورات بسیار مفید است:

i	i<=10	عملیات
1	yes	چاپ ۱
2	yes	چاپ ۲
3	yes	چاپ ۳
4	yes	چاپ ٤
5	yes	چاپ ٥
6	yes	چاپ ٦
7	yes	چاپ ۷
8	yes	چاپ ۸
9	yes	چاپ ۹
10	yes	چاپ ۱۰
11	no	خروج از حلقه

شما با ترسیم چنین جدولی بر روی کاغذ و ردگیری گام به گام حلقه قادر به نوشتن حلقه های پیچیده تر در C++خواهید بود، با کمی تمرین به راحتی به این مهم خواهید رسید.

مثال) با استفاده از حلقه تکرار for کدی بنویسید که در آن کاربر جمله ای را وارد کرده و در انتهای جمله Enterرا فشار دهد و برنامه تعداد حروف جمله را نمایش دهد:

```
1  int  count;
2  cout << "Enter a statement with enter in end:" << endl;
3  for ( count = 0; cin.get()!='\r' ; count++ );
4  cout << "Lenght of statement is: " << count;
5  getch();
```

```
Enter a statement with enter in end:
I like C++ programming!
Lenght of statement is: 23
```

در برنامه بالا چند نکته مهم وجود دارد که به بررسی آنها می پردازیم:

الف) به شرط حلقه توجه کنید. تابع cin.get() برای تشخیص ورود Enter بکار می رود. تابع دانه دانه حروف ورودی را بررسی می کند و به محض Enter کردن کاربر از حلقه خارج می شود.

شرط حلقه به این معنی است که : تا زمانیکه حرف وارد شده کاربر مخالف Enter (\r) است حلقه را ادامه بده و به count هر بار یک واحد اضافه کن.

ب) برای حلقه هیچ بلوکی از دستورات وجود ندارد و در انتهای آن هم از سمی کالن استفاده کردیم. در واقع با اینکار ما فقط خواستیم که تعداد حروف جمله را شمارش کنیم.

حلقه های تودرتو

گاهی اوقات لازم است که در یک حلقه، یک یا چند حلقه دیگر هم استفاده نمود.

معروفترین مثال برنامه نویسی در مورد حلقه های تودرتو، نمایش جدول ضرب اعداد است:

```
1  int  i,j;
2
3  for ( i = 0; i<=10 ; i++ )
4  {
5     for ( j = 0; j<=10 ; j++ )
6        cout <<  i*j  << "\t";
7     cout << endl;
8  }
```

```
1  2  3  4  5  6  7  8  9  10
2  4  6  8  10 12 14 16 18 20
3  6  9  12 15 18 21 24 27 30
4  8  12 16 20 24 28 32 36 40
5  10 15 20 25 30 35 40 45 50
6  12 18 24 30 36 42 48 54 60
7  14 21 28 35 42 49 56 63 70
8  16 24 32 40 48 56 64 72 80
9  18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100
```

در ابتدا، برنامه وارد حلقه اول شده و شرط را بررسی می کند و با درستی آن به اجرای دستورات حلقه می پردازد. اینجا برای حلقه آکولاد وجود دارد پس تمامی دستورات درون بلوک به ترتیب اجرا می شوند.

سطر بعدی هم یک حلقه است یعنی حلقه جاری به ازای تعداد تکرار حلقه اول باید تکرار شود و در هر تکرار از حلقه بالا به تعداد تکرار خود نیز تکرار می شود، شرط آن بررسی شده و با درستی شرط به انجام دستورات حلقه می پردازد. به دلیل عدم وجود آکولاد برای این حلقه، تنها سطر بعدی دستور حلقه داخلی است و تا نقیض شدن شرط حلقه آن سطر اجرا می شود. در این مثال ۱۰ بار حلقه درونی تکرار می شود و با هر بار تکرار فاصله ای بعد از نمایش عدد مورد نظر هم قرار می دهد.

سپس برنامه با اجرای دستور بعدی به خط جدید می رود. از حلقه اول ۹ بار تکرار دیگر مانده، پس این رویه تکرار شده تا برنامه کامل گردد. در آخر وقتی i برابر با ۱۱ می شود از حلقه بیرونی خارج شده و برنامه به پایان می رسد.

مثال مهم) قصد داریم با استفاده از حلقه های تکرار تودرتو شکل زیر را ایجاد نماییم:

```
*****
****
***
**
*
```

روش کلی کار به این صورت است که به ازای تعداد سطرها به یک حلقه تکرار for بیرونی و به ازای تعداد ستونها به یک حلقه for درونی نیاز داریم. چون از ابتدا به انتها از تعداد ستاره ها کم می شود پس باید حلقه های خود را کاهشی بنویسیم:

```
for ( int i = 5; i>0 ; i-- )
{
    for ( int j = i; j>0 ; j-- )
        cout << "*";
    cout << "\n";
}
```

با شروع حلقه for اول، برنامه وارد بلوکی می شود که دارای حلقه for دیگری است و دو دستورcout ، که اولین دستور مربوط به حلقه درونی است و دومی مربوط به حلقه بیرونی.

متغیر i از ٥ شروع شده با هر بار اجرای حلقه یک واحد کاهش میابد. به این دلیل متغیر j را برابر با متغیر i گرفته ایم چون با کاهش i آنهم باید کاهش یابد یعنی در هر بار چرخش حلقه بیرونی (هر سطر).

در ابتدا i برابر با ٥ است و شرط حلقه هم درست است پس وارد دستورات حلقه for بیرونی می شود و به حلقه for درونی برخورد می کند که j برابر با i یعنی ٥ است و شرطش هم درست است پس ٥ بار ستاره چاپ می کند و با نقیض شدن شرط حلقه) وقتی که j برابر صفر می شود (از حلقه درونی خارج می شود و ادامه دستورات حلقه for بیرونی را انجام میدهد یعنی یک سطر پایین می رود.

این بار یک واحد از j کم شده و برابر با ٤ می شود که باز هم شرط حلقه بیرونی درست است و دستورات آنرا دوباره اجرا می کند. در درون حلقه for درونی j برابر با i یعنی ٤ شده و شرطش هم درست است پس اینبار چهار ستاره چاپ می کند و به همان ترتیب قبل دستورات ادامه میابد تا شکل بالا حاصل می شود.

مثال) می خواهیم شکل زیر را با استفاده از دستور حلقه تکرار for تودرتو در برنامه نویسی C++رسم نماییم:

```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

برای رسم این شکل ما ابتدا شکل را به ٢ قسمت تقسیم کرده و برای رسم هر قسمت از دستور حلقه تکرار forتودرتو استفاده می کنیم:

```
for ( int i = 6; i>0 ; i-- )
{
   for ( int j = i; j>0 ; j-- )
      cout << " ";

   for ( j = i; j<6 ; j++ )
      cout << "*";

   for ( j = i; j<=6 ; j++ )
      cout << "*";

   for ( int j = i; j>0 ; j-- )
      cout << " ";

   cout << "\n";
}

for ( i = 6; i>=0 ; i-- )
{
   for ( j = i; j<6 ; j++ )
      cout << " ";

   for ( j = i; j>0 ; j-- )
      cout << "*";

   for ( j = i; j>=0 ; j-- )
      cout << "*";

   for ( int j = i; j<6 ; j-- )
      cout << " ";

    cout << "\n";
}
```

عزیزان اگر کمی دقت کنید و با حوصله کد بالا را بررسی نمایید حتما مطلب را یاد گرفته و با کمی تمرین براحتی می توانند در C++دستورات حلقه تکرار for تو در توی پیچیده را براحتی حل نمایید.

ساختارهای کنترلی - حلقه - while

در این فصل از آموزش برنامه نویسی C++ قصد داریم به بررسی ساختار حلقه while() بپردازیم که وظیفه ای شبیه به حلقه for()را برعهده دارد.

```
while ( عبارت شرطی )
{
   Line Command 1;
   Line Command 2;
   Line Command n;
}
```

از این ساختار برای ایجاد حلقه های تکرار استفاده می شود و تا زمانیکه عبارت شرطی داخل پرانتز while() درست باشد دستورات مربوطه اجرا خواهند شد و به محض نادرستی شرط، کنترل دستورات از حلقه خارج خواهد شد.

```
1  int n;
2
3  cout << "Enter the starting number that bigger than zero : " ;
4  cin >> n ;
5
6  while ( n > 0 )
7  {
8     cout << n << ", " ;
9     n--;
10 }
11
   cout << "Fire! \n" ;
```

```
Enter the starting number that bigger than zero : 8
8, 7, 6, 5, 4, 3, 2, 1, Fire!
```

کد برنامه نویسی بالا، عددی را از کاربر گرفته و با استفاده از حلقه while شمارش معکوس آنرا در خروجی نمایش می دهد.

در برنامه C++بالا، اگر عدد صفر یا کمتر از آنرا وارد کنیم شرط حلقه while نادرست بوده و دستورات درون حلقه اجرا نخواهد شد و فقط عبارت Fire!چاپ می شود.

شکل دیگری از این دستور وجود دارد که شبیه به while()بوده و به آن حلقه do while گفته می شود، با این تفاوت که اگر شرط درون آن نادرست باشد دستورات درون حلقه حداقل یکبار اجرا می شوند و اگر شرط درست باشد، حلقه تا زمان نادرستی شرط ادامه خواهد یافت و ساختار آن بدینگونه است:

```
do
{
   Line Command 1;
   Line Command 2;
   Line Command n;
}
while ( عبارت شرطی )
{
   Line Command 1;
   Line Command 2;
   Line Command n;
}
```

در ساختار do while()ابتدا دستورات doاجرا شده و بعد شرط درون while()بررسی می شود و در صورت درستی شرط، اجرای دستورات doادامه پیدا خواهد کرد.

نکته) دوستان توجه داشته باشند که اگر دستورات مربوط به for, while, do, ... فقط یک دستور باشد نیازی به قرار دادن دستور درون آکولاد نیست و تکرار فقط بر روی تنها سطر دستور ادامه خواهد داشت و تا زمانیکه شرط برقرار باشد فقط و فقط آن سطر تکرار و اجرا می شود و تا زمان پایان به خط بعدی نخواهد رفت اما اگر دستورات بیش از یکی باشد لازم است که آنها را در آکولاد قرار دهیم و این قانون کلی در برنامه نویسی است.

```
1  int n;
2
3  do
4  {
5     cout << "Enter number (0 to end): " ;
6     cin >> n ;
      cout << "You entered: " << n << "\n" ;
```

```
7   } while ( n != 0 )
8
```

```
Enter number (0 to end): 1298
You entered: 1298
Enter number (0 to end): 35
You entered: 35
Enter number (0 to end): 0
You entered: 0
```

اگر عدد ورودی در کد بالا در ابتدا صفر باشد دستورات یکبار اجرا می شود.