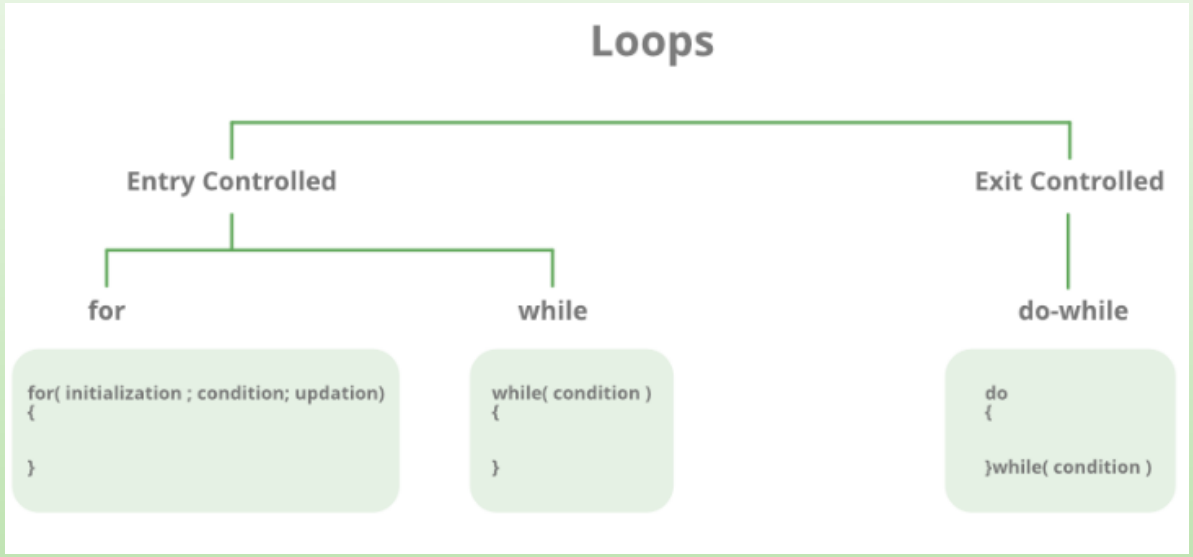
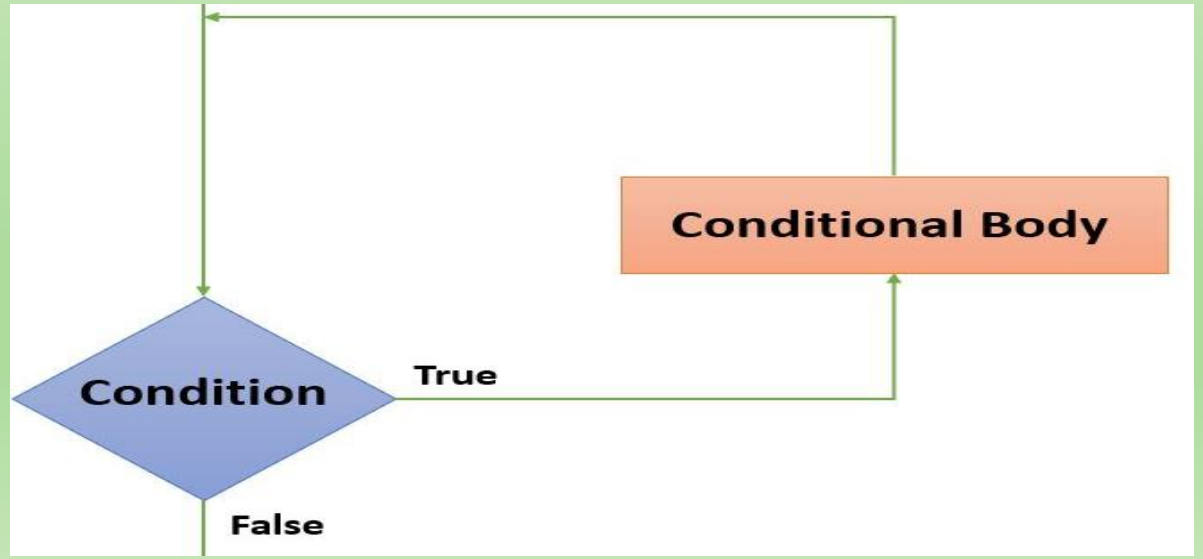
DÖNGÜ YAPILARI

Hazırlayan : Muhammed Esad Goncaloğlu

DÖNGÜ YAPILARI

- Döngü, belirli bir koşul sağlanana kadar aynı işlemin birden çok kez tekrarlanması olarak tanımlanabilir.
- C/C++ dilinde kullanılan üç tür döngü vardır.
- Bunlar for, while ve do-while döngü yapılarıdır.
- Kodun yeniden kullanılabilirliğini sağlar.
- Döngüleri kullanarak aynı kodu tekrar tekrar yazmamıza gerek kalmaz.
- Döngüleri kullanarak, veri yapılarının (dizi veya bağlantılı listeler) öğeleri arasında gezinebiliriz





FOR DÖNGÜ YAPISI

- C/C++'da en çok kullanılan döngüdür.
- Bu döngü, bir ifadeyi birkaç kez yinelemek için tasarlanmıştır.
- For döngüsünün üç değer alan kısmı vardır.
- Başlatma, bir for döngüsünde yalnızca bir kez gerçekleştirilir.
- Koşul, her yineleme için sabit bir değeri test eden ve onunla karşılaştıran ifadeler kümesidir. Koşul sonuçları yanlışsa yinelemeleri durdurur.
- Artırma veya eksiltme, koşul değerini ihtiyacına göre değiştirir.

```
for (initialization; condition; increment/decrement)
{
// statement to be executed
}
```

```
#include <iostream>
using namespace std;
int main(){
    int i;
    for(i=0;i<5;i++){
        cout << i << "\t";
    }
    return 0;
}
```

```
C:\Users\ESAD\Desktop\Untitled1.exe
0       1       2       3       4
--------------------------------
Process exited after 0.03356 seconds with return value 0
Press any key to continue . . .
```

WHILE DÖNGÜ YAPISI

- En temel döngü türüdür ve diğer döngü yapılarına göre daha basit bir yapısı vardır.
- Parantez bloğunun içerisindeki durumu değerlendirir.
- Önce koşulu kontrol eder ve doğruysa programın döngü gövdesinin içine girmesine izin verir ve koşul başarısız olduğunda program döngüden çıkar.
- Genelde arttırma ve eksiltme operatörleri ile beraber çalışır.

```
while (expression)
{
  // statement to be executed or repeated
}
```

```
#include <iostream>
using namespace std;
int main(){
    int i=0;
    while(i<5){
        cout << i << "\t";
        i++;
    }
    return 0;
}
```

```
C:\Users\ESAD\Desktop\Untitled1.exe
0       1       2       3       4
--------------------------------
Process exited after 0.03633 seconds with return value 0
Press any key to continue . . .
```

DO-WHILE DÖNGÜ YAPISI

- ‘while’ döngüsüne çok benzerdir.
- Tek farkı koşulun en son kontrol edilmesidir.
- Koşul döngü gövdesinin sonunda bildirildiği için bloğun içindeki ifade yanlış olsa bile en az bir kere çalıştırır.
- İşlem ‘do’ kısmında gerçekleşir. ‘while’ kısmında sadece koşul sağlaması yapılır. Koşul sağlandıkça döngü çalışır. Sağlanmadığında döngü biter.

```
do
{
// statement to be executed or repeated
}
while (condition);
```

```
#include <iostream>
using namespace std;
int main(){
    int i=0;
    do{
        i++;
        cout << i << "\t";
    }while(i<5);
    return 0;
}
```

```
C:\Users\ESAD\Desktop\Untitled1.exe
1       2       3       4       5
--------------------------------
Process exited after 0.03104 seconds with return value 0
Press any key to continue . . .
```

BREAK VE CONTINUE DEYİMLERİ

- Genellikle döngülerde ve ‘switch-case’ yapılarında kullanılır.
- ‘break’ deyimi, bir döngüden çıkmak için kullanılır. ‘switch-case’ de kullanımı ise sadece ilgili ‘case’ bölümünün çalıştırılması içindir.
- ‘continue’ deyimi ise, belirtilen bir koşul oluşursa bir yinelemeyi (döngüde) keser ve döngüdeki bir sonraki yinelemeyle devam eder.

```cpp
#include <iostream>
using namespace std;
int main(){
    int i=0;
    for(i=0;i<5;i++){
        if(i==3){
            break;
        }
        cout << i << "\t";
    }
    return 0;
}
```

```
C:\Users\ESAD\Desktop\Untitle
0       1       2
--------------------------------
Process exited after 0.0358
Press any key to continue .
```

```cpp
#include <iostream>
using namespace std;
int main(){
    int i=0;
    for(i=0;i<5;i++){
        if(i==3){
            continue;
        }
        cout << i << "\t";
    }
    return 0;
}
```

```
C:\Users\ESAD\Desktop\Untitled1.
0       1       2       4
--------------------------------
Process exited after 0.03446
Press any key to continue . .
```

İÇ İÇE DÖNGÜ YAPILARI

- Döngülerin iç içe geçmesi, ifadelerin başka bir döngü içinde döngüye alınmasına izin veren özelliktir.
- Başka bir döngü içinde herhangi bir sayıda döngü tanımlanabilir, yani herhangi bir sayıda döngü tanımlamak için herhangi bir kısıtlama yoktur.
- Genellikle diziler ile beraber kullanılır.
- Herhangi bir döngü türünü başka bir döngü içinde tanımlanabilir.

```
#include <iostream>
using namespace std;
int main(){
    int i,j;
    for(i=0;i<5;i++){ // 1st loop
        for(j=0;j<5;j++){ // 2nd loop
        cout << (i*5)+j+1 << "\t"; // rows
}
cout << endl; // end of row
}
    return 0;
}
```

```
1       2       3       4       5
6       7       8       9       10
11      12      13      14      15
16      17      18      19      20
21      22      23      24      25
```

KAYNAKÇA

- https://www.javatpoint.com/c-loop
- https://usemynotes.com/what-is-loop-statement-in-c/
- https://www.w3schools.com/c/c_break_continue.php
- https://www.javatpoint.com/nested-loops-in-c