



Loops in C++ Conditional Loops & Counter Loops



اَللّٰهُمَّ ارْزُقْنِيْ عِلْمًا نَّافِعًا وَاسِعًا عَمِيْقًا

اَللّٰهُمَّ ارْزُقْنِيْ رِزْقًا وَّاسِعًا حَالًا لَا طَيِّبًا
مُّبَارَكًا مِنْ عِنْدِكَ

Revision: Say Welcome to UET

Let's say we want to write **Welcome to UET** on the Console.



```
#include<iostream>
using namespace std;

main(){

    cout<<"Welcome to UET!!";

}
```



| Say Welcome to UET 5 times

Let's say we want to write Welcome to UET on the Console 5 times.

```
#include<iostream>
using namespace std;

main(){

    cout<<"Welcome to UET!!";

}
```



Code Repetition

What if we want to write **Welcome to UET** on the Console **100** times or **1000** times ???

```
#include<iostream>
using namespace std;
main(){
    cout<<"Welcome to UET!!" << endl;
    cout<<"Welcome to UET!!" << endl;
    cout<<"Welcome to UET!!" << endl;
    cout<<"Welcome to UET!!" << endl;
    cout<<"Welcome to UET!!" << endl;
}
```

Code Repetition: Problem

Is there a way, we don't have to **repeat** the same instructions again and again?



Code Repetition: Solution

In High level languages, **Loops** are used to repeat the same command without writing it multiple times.



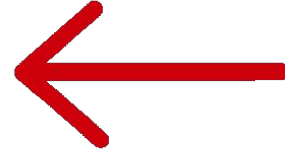
loop



Loops

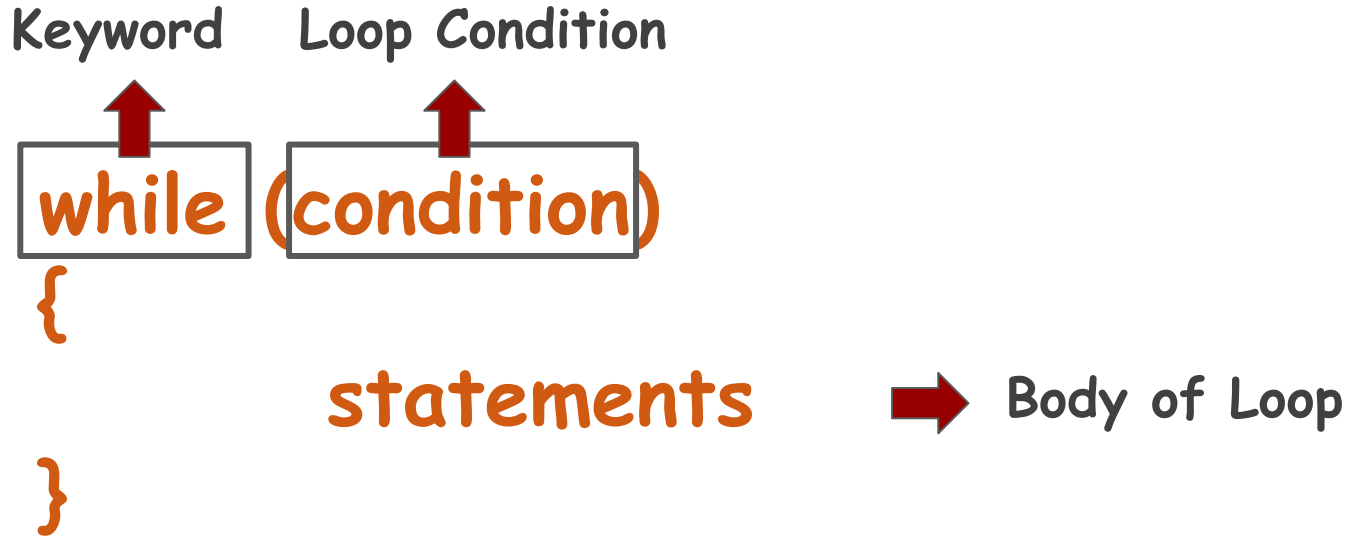
There are 2 types of Loops.

1. **Conditional** Loops
2. **Counter** Loops



Conditional Loops

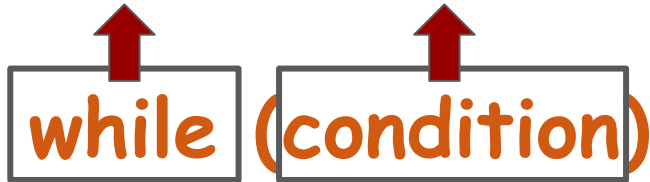
In C++, we have while loop.



Conditional Loops

Can we make this while Loop to run for 5 times?

Keyword Loop Condition


while (condition)

{

statements

}



Body of Loop

Task:

We want to write
Welcome to UET
on the Console **5**
times.

Conditional Loops

We want to write **Welcome to UET** on the Console 5 times.

```
#include <iostream>
using namespace std;
main()
{
    int count = 0;
    while (count < 5)
    {
        cout << "Welcome to UET!!" << endl;
        count = count + 1;
    }
}
```

Conditional Loops

We want to write **Welcome to UET** on the Console 5 times.

Initialize
the loop
counter

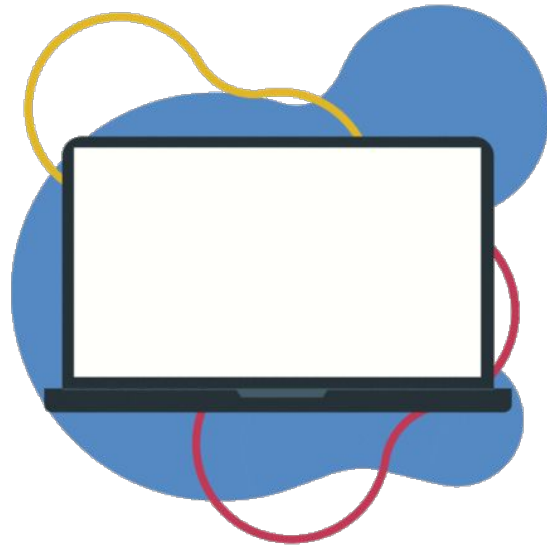
Check
the
Condition

```
#include <iostream>
using namespace std;
main()
{
    int count = 0;
    while (count < 5)
    {
        cout << "Welcome to UET!!" << endl;
        count = count + 1;
    }
}
```

Update
the Loop
counter

Conditional Loop: Working Example

Suppose, the requirement is to **keep taking numbers as input** from the user until the user enters **-1**.



Conditional Loop: Working Example

Output on the **Console** is as follows.

```
C:\C++ Programming\Level 1>c++ program.cpp -o program.exe

C:\C++ Programming\Level 1>program.exe
Please Enter any Number or -1 to Exit: 4
Please Enter any Number or -1 to Exit: 3
Please Enter any Number or -1 to Exit: 8
Please Enter any Number or -1 to Exit: -1
Program Ends

C:\C++ Programming\Level 1>
```

Conditional Loop: Working Example

Suppose, the requirement is to **keep taking numbers as input** from the user until the user enters **-1**.

```
#include<iostream>
using namespace std;
main(){
    int number = 0;
    while (number != -1)
    {
        cout << "Please Enter any Number or -1 to Exit: ";
        cin >> number;
    }
    cout << "Program Ends";
}
```


Conditional Loop: Working Example

Suppose, the requirement is to **keep taking numbers as input** from the user until the user enters **-1**.

```
#include<iostream>
using namespace std;
main(){
    int number = 0;
    while (number != -1)
    {
        cout << "Please Enter any Number or -1 to Exit: ";
        cin >> number;
    }
    cout << "Program Ends";
}
```

We are executing the loop until a certain condition is met therefore, it is called **Conditional Loop**

Conditional Loop

Now, can we initialize, check the condition and update the loop counter in a single line?

Initialize
the loop
counter

Check
the
Condition

```
#include <iostream>
using namespace std;
main()
{
    int count = 0;
    while (count < 5)
    {
        cout << "Welcome to UET!!" << endl;
        count = count + 1;
    }
}
```

Update
the Loop
counter

Loops

There are 2 types of Loops.

1. **Conditional** Loops
2. **Counter** Loops



Counter Loop

In **Counter** Loop, the program knows beforehand about **how many times** a specific instruction or set of instructions will be executed.

Keyword	Initial Statement	Loop Condition	Update statement
<div>for</div>	<div>(int x = 0;</div>	<div>x < 5;</div>	<div>x = x + 1</div>
{			
	//body	➡	Body of Loop
}			

Counter Loop: Working Example

Let's see how to write **Welcome to UET** on the Console **5** times using for loop.

```
#include<iostream>
using namespace std;

main()
{
    for(int count = 0; count < 5; count = count + 1)
    {
        cout << "Welcome to UET!!" << endl;
    }
}
```

Counter Loop: Working Example

Output on the **Console** is as follows.

```
C:\C++>c++ example.cpp -o example.exe
```

```
C:\C++>example.exe
```

```
Welcome to UET!!
```

```
Welcome to UET!!
```

```
Welcome to UET!!
```

```
Welcome to UET!!
```

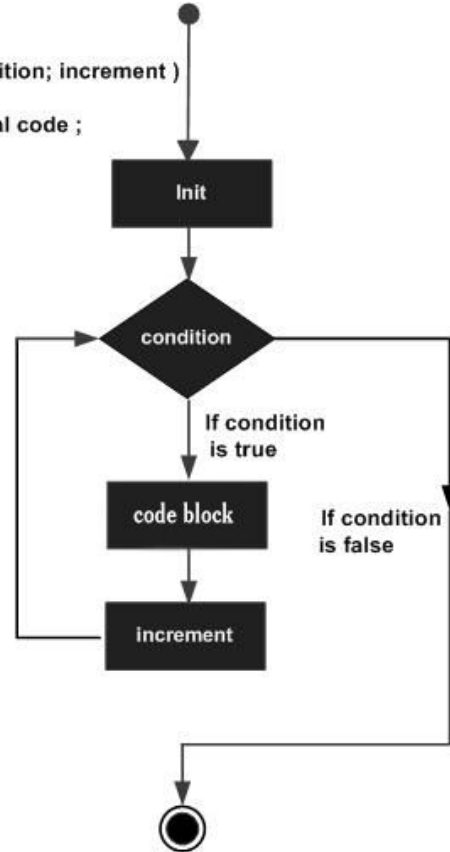
```
Welcome to UET!!
```

```
C:\C++>
```




Counter Loop: Flow of for Loop

```
for( init; condition; increment )  
{  
    conditional code ;  
}
```



Counter Loop: Working Example

Now, Let's say we have to write **Welcome to UET** on the Console **20** times using for loop.



In this **Code Snippet**, how many places the **changes** will have to be done?

```
#include<iostream>
using namespace std;

main(){

    for(int x = 0; x < 5; x = x + 1)
    {
        cout << "Welcome to UET!!" << endl;
    }
}
```

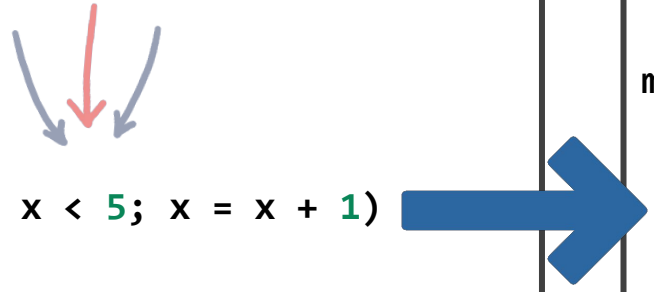

Counter Loop: Working Example

In this **Code Snippet**, how many places the **changes** will have to be done?

```
#include<iostream>
using namespace std;

main(){

    for(int x = 0; x < 5; x = x + 1)
    {
        cout << "Welcome to UET!!" << endl;
    }
}
```



```
#include<iostream>
using namespace std;

main(){

    for(int x = 0; x < 20; x = x + 1)
    {
        cout << "Welcome to UET!!" << endl;
    }
}
```

Counter Loop: Working Example

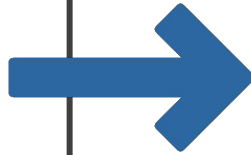
This code Snippet, will print **Welcome to UET!!** on the console **20** times.

```
#include<iostream>

using namespace std;

main(){

    for(int x = 0; x < 20; x = x + 1)
    {
        cout << "Welcome to UET!!" << endl;
    }
}
```

[illegible]

Learning Objective

In this lecture, we learnt how to write a **C++ Program** that **repeats** a Set of Instructions for a **specific number of times** to solve the given problem using **Counter Loop**.



Learning Objective

In this lecture, we learnt how to write a **C++ Program** that **repeats** a Set of Instructions for an **unknown number of times** to solve the given problem using **Conditional Loop**.



Conclusion

- Loops are of 2 types
 1. Counter Loops.
 2. Conditional Loops
- In Counter Loop, the program knows beforehand about how many times a specific instruction or set of instructions will be executed.
- In C++ programming language, for loop is used as a counter loop.



Conclusion

- **Conditional** loops help to repeat a set of instructions until some condition is **true**.
- There are **two** common places for it use.
 1. **Reading an unknown** amount of input from the user
 2. **Validating** input.
- **C++** provides a **while loop** that is used as a conditional loop.



Self Assessment

1. What is the **output**?

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

2. What is the **output** of the following code fragment?

```
for ( int j = 10; j > 5; j-- )  
{  
    cout << j << " ";  
}
```



Self Assessment

4. What is the **output**?

```
int number = 6;
while (number > 0)
{
    cout << number << " ";
    number -= 3;
}
```

5. What is the **output** of the following code fragment?

```
int number = 4;
while (number >= 0){
    number = number -1;
    cout << number << endl;
}
```



Self Assessment

3. Write a **C++ Program** that asks the user to enter **5** numbers, one at a time, and add them together. This is called a **Running Total**. Once the user is done, display the total sum on the Console.



Self Assessment

6. Write a program to keep asking for a number until the user enters a **negative number**. At the end, print the sum of all entered numbers.

7. Write a program to ask for a name until the user enters **"END"**. Print the hello with the name each time when the user enters. At the end of the program, print **"END"** when the user enters **END**.

