



Week 7

Session 7: Repetitive tasks using 'while' and 'do-while' loop statements

Element 4: Plan and execute C++ applications using loop structures, 'for' and 'while' statements

ECT 124: Writing Programs using C++



Performance criteria (PC) for E4



PC1: Write code for repetitive tasks using 'for' loops.

PC2: Write code for repetitive tasks using 'while' and 'do-while' statements.

In this lesson!



Learning objectives:

By the end of this lesson, the student should be able to:

- ✓ Write applications in C++ using 'while' and 'do-while' loop statements.



Class Activity 1

Drag and Drop

- for loop statement
- while loop statement
- do-while loop statement

Drag the correct loop descriptor according to the loop types.

^ Instructions

Entry controlled loops

Exit controlled loops

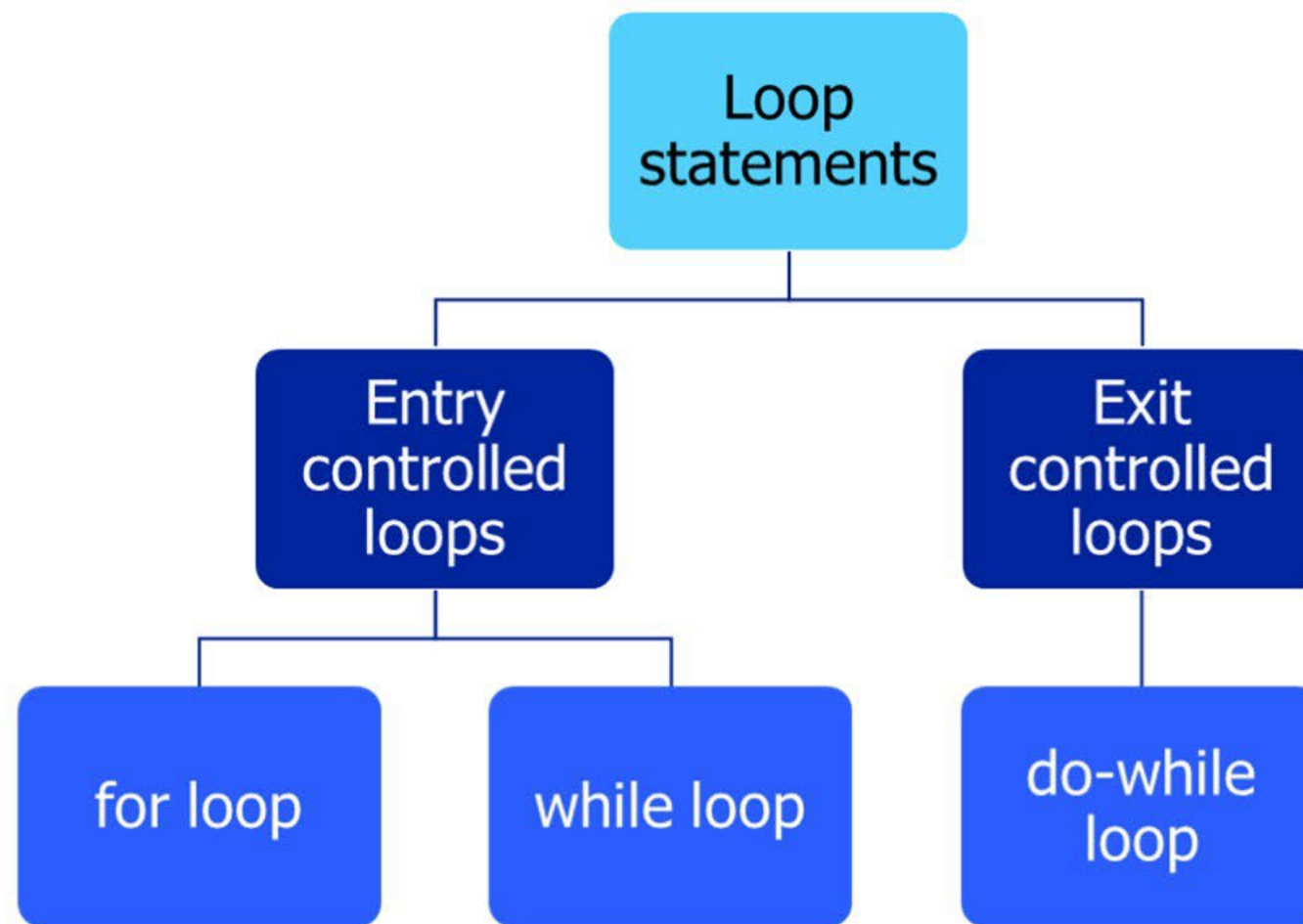


Submit



Loop Statements

- The while loop is an entry controlled loops whereas do-while loop is an exit controlled loops.





The while Loop Statements

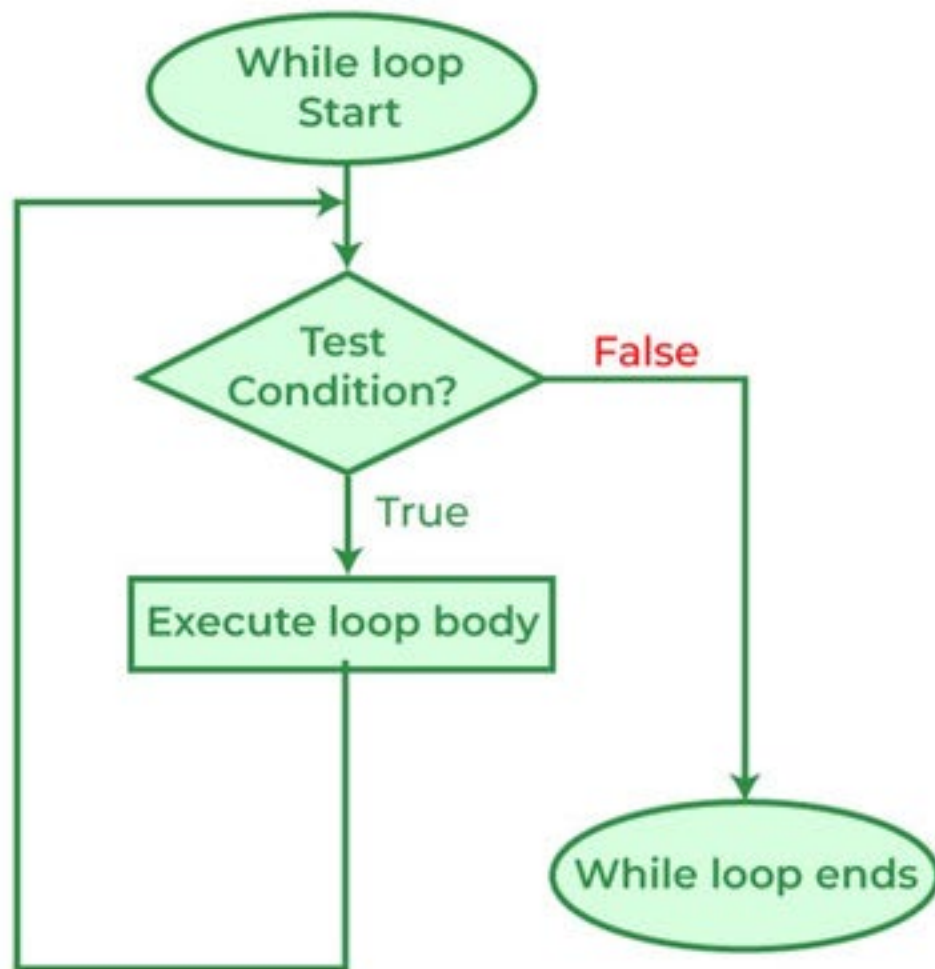
- The **while loop** in C++ is used when we do not know the exact number of iterations of the loop beforehand. The loop execution is terminated on the basis of the test condition.
- Unlike for loop, the number of times the loop body is needed to be executed is known before.
- The while loop statement is made of the reserved word **while**, followed by a Boolean expression (called the condition) in parentheses, followed by a single or multiple lines of statement, which is the body of the while statement. The syntax for while loop is:

```
while (test_condition)
{
// statement -1;
// statement -n;
update_expression
}
```



test_condition	Testing the condition. If the condition is true, we will execute the body of the loop and go to update expression. Otherwise, we will exit from the while loop.
update_expression	After executing the loop body, this expression increments/decrements the loop variable by some value.
Body	This is a group of statements

- The flowchart and execution flow of while loop are described below:



How does a While loop execute?

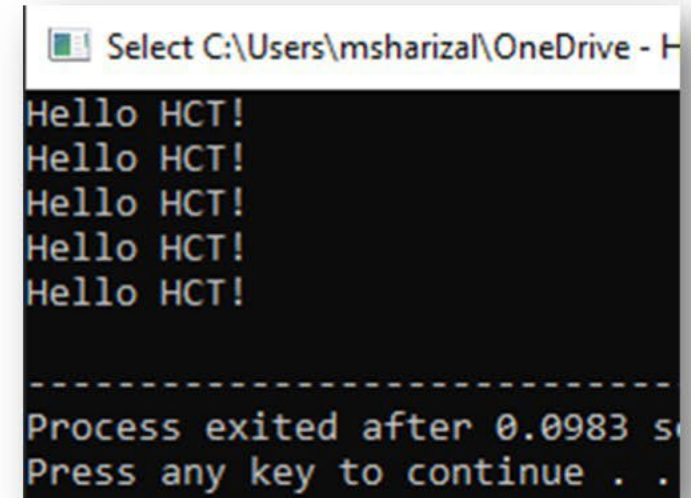
1. Control falls into the while loop.
2. The flow jumps to Condition
3. Condition is tested.
 - If the Condition yields true, the flow goes into the Body.
 - If the Condition yields false, the flow goes outside the loop
4. The statements inside the body of the loop get executed.
5. Updation takes place.
6. Control flows back to Step 2.
7. The while loop has ended and the flow has gone outside.



- The while loop should end at some point, or become false at some point. Otherwise, the loop will continue looping forever.
- The while loop repeats statement while expression is true. If, after any execution of statement, expression is no longer true, the loop ends, and the program continues right after the loop.
- Statement may be a single statement or a block of statements. The condition may be any expression, and true is any non-zero value. The loop iterates while the condition is true.
- The while loop might not ever run. When the condition is tested and the result is false, the loop body will be skipped and the first statement after the while loop will be executed.

Example 1 Print "Hello HCT" 5 times depending on condition.

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int i = 1;
7
8      while (i < 6)    // test condition
9      {
10         cout << "Hello HCT!" << endl;
11
12         i++; // update expression
13     }
14     return 0;
15 }
```



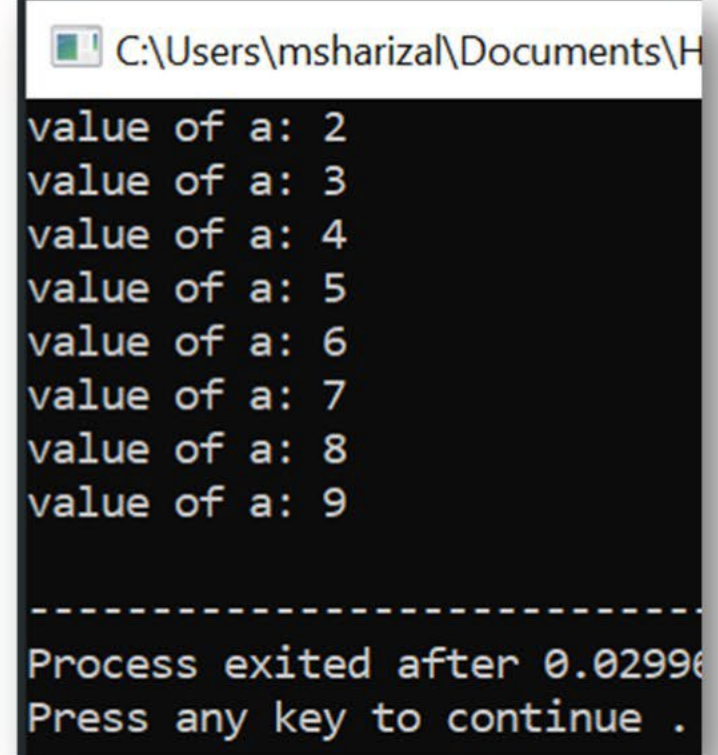
Select C:\Users\mscharizal\OneDrive - H

Hello HCT!
Hello HCT!
Hello HCT!
Hello HCT!
Hello HCT!

Process exited after 0.0983 s
Press any key to continue . .

Example 2 Print numbers from 2 to 9.

```
1  #include <iostream>
2  using namespace std;
3
4  int main ()
5  {
6      int a = 2;
7
8      while( a < 10 )
9      {
10         cout << "value of a: " << a << endl;
11         a++;
12     }
13     return 0;
14 }
```



```
C:\Users\msharizal\Documents\H
value of a: 2
value of a: 3
value of a: 4
value of a: 5
value of a: 6
value of a: 7
value of a: 8
value of a: 9
-----
Process exited after 0.02996
Press any key to continue .
```

The counter can be represented also by a variable letter, like **a** in this case and initialized with non-zero number (which is 2).



Example 3

Print numbers with TRUE and FALSE conditions.

```
1  #include <iostream>
2  using namespace std;
3  int main ()
4  {
5  // Declaration of the limit and counter
6  int n, count;
7
8  // Input the value of n (limit)
9  cout << "Enter the number of integers to print: ";
10 cin >> n;
11
12 // Printing integers
13 count = 0;
14 while (count < n)
15 {
16 cout << count << endl;
17 count++;
18 }
19 return 0;
20 }
```

```
C:\Users\mscharizal\OneDrive - Higher Colleges of Te
Enter the number of integers to print: 4
0
1
2
3
-----
Process exited after 5.951 seconds with r
Press any key to continue . . .
```

```
C:\Users\mscharizal\OneDrive - Higher Colleges of Te
Enter the number of integers to print: 0
-----
Process exited after 2.486 seconds with r
Press any key to continue . . .
```

```
C:\Users\mscharizal\OneDrive - Higher Colleges of Tech
Enter the number of integers to print: -4
-----
Process exited after 5.17 seconds with retu
Press any key to continue . . .
```

Note that in the last two runs the condition (count < n) is false in line 14 and the program never enters the body of the loop.



Example 4

Print the square root of each number input by the user.

```
1  #include <iostream>
2
3  #include<cmath> /*A pre-defined header file for math functions
4  (we are using square root function for this program)*/
5
6  using namespace std;
7
8  int main()
9  {
10     int x;
11     cout << "Enter a positive number: ";
12     cin >> x;
13
14     while (x > 0)
15     {
16         cout << "sqrt (" << x << ") = " << sqrt(x) << endl;
17         cout << "\nEnter another positive number (or 0 to quit): ";
18         cin >> x;
19     }
20
21     return 0;
22 }
```

C:\Users\msharizal\OneDrive - Higher Colleges of Technology\MyDoc

```
Enter a positive number: 100
sqrt (100) = 10

Enter another positive number (or 0 to quit): 25
sqrt (25) = 5

Enter another positive number (or 0 to quit): 0

-----
Process exited after 7.249 seconds with return val
Press any key to continue . . .
```

- A pre-defined cmath header file is used to enable the usage of sqrt function for the program.
- Since the condition defined is $x > 0$, and input number of 0 will terminate the loop (false condition)



The do-while Loop Statements

- The **do-while loop** behaves like a while loop, except that condition is tested after the execution of statement instead of before, ensuring at least one execution of loop, even if condition is never fulfilled.
- Thus, the do-while loop is an exit controlled loop whereas the other two loops (for loop and while loop) are entry controlled loops.
- The do-while loop is preferred over a while-loop when the statement needs to be executed at least once, such as when the condition that is checked is within the loop statement itself.
- Unlike while loops which test the loop condition at the top of the loop, the do-while loop checks its **condition at the bottom of the loop**. Thus, the body of the loop in a **do-while loop is always executed at least once**.

We use a *do-while* loop when the problem requires that the body of the loop be executed at least once.



- The syntax for do-while loop is:

```
do
{
// statement -1;
// statement -n;
update_expression
}
while (test_condition);
```

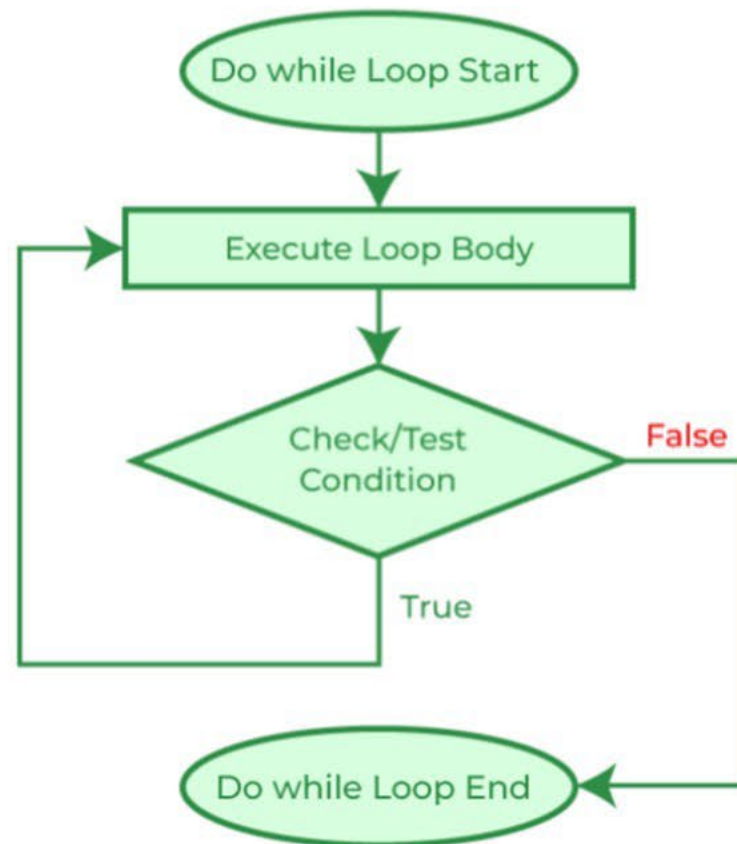


test_condition	Testing the condition. If the condition is true, we will execute the body of the loop and go to update expression. Otherwise, we will exit from the while loop.
update_expression	After executing the loop body, this expression increments/decrements the loop variable by some value.
Body	This is a group of statements

- The do-while loop consists of five parts: the reserved word **do**, the statement that is the body of the loop (usually a compound statement enclosed in braces), the reserved word **while**, a logical expression in parentheses, and a semicolon.
- Note that the **semicolon is required at the end** of the do-while loop.



- The flowchart and execution flow of do-while loop:



How does a do-While loop execute?

1. Control falls into the do-while loop.
2. The statements inside the body of the loop get executed.
3. Updation takes place.
4. The flow jumps to Condition
5. Condition is tested.
 - If the Condition yields true, go to Step 6.
 - If the Condition yields false, the flow goes outside the loop
6. The flow goes back to Step 2.
7. The do-while loop has been ended and flow has gone outside the loop.

- Notice that the conditional appears at the end of the loop, so the statement(s) in the loop execute once before the condition is tested.
- If the condition is true, the flow of control jumps back up to the loop again. It repeats until the condition becomes false.



Example 5

Print "Hello HCT" several times depending on condition.

```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     int i = 1;
7
8     do
9     {
10         cout << "Hello HCT\n";
11
12         i++; // Update expression
13     }
14     while (i < 5); // Test expression
15
16     return 0;
17 }
```

```
C:\Users\mshariza\O
Hello HCT
Hello HCT
Hello HCT
Hello HCT
-----
Process exited aft
Press any key to c
```

Printed 4 line of outputs for the TRUE case where $1 < 5$.

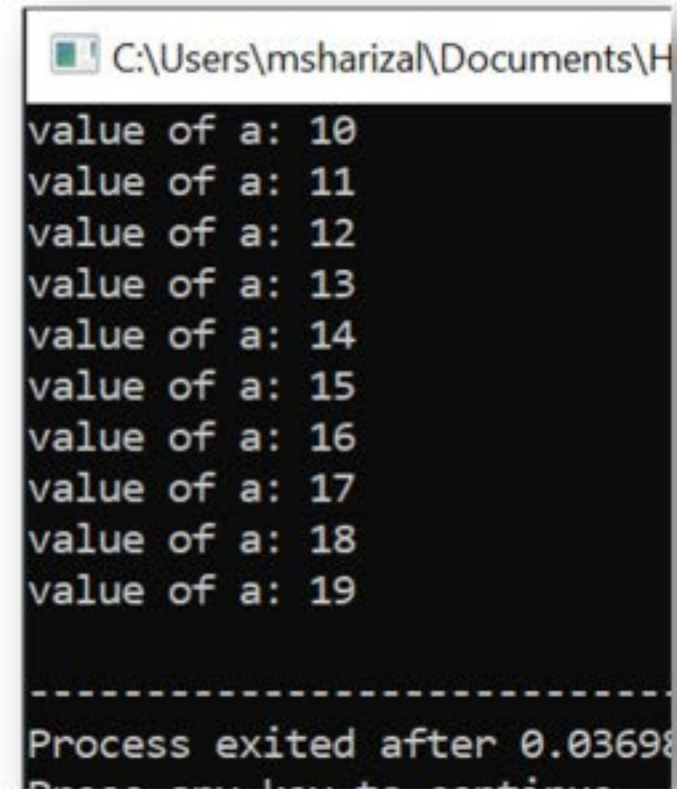
```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     int i = 10;
7
8     do
9     {
10         cout << "Hello HCT\n";
11
12         i++; // Update expression
13     }
14     while (i < 5); // Test expression
15
16     return 0;
17 }
```

```
C:\Users\mshariza
Hello HCT
-----
Process exited
Press any key to
```

Printed only 1 line of output for the FALSE case where $10 > 5$. The loop is at least performed 1 time for a FALSE case.

Example 6 Print numbers from 10 to 19.

```
1  #include <iostream>
2  using namespace std;
3
4  int main ()
5  {
6  int a = 10;
7
8  do
9  {
10 cout << "value of a: " << a << endl;
11 a++;
12 }
13 while( a < 20 );
14
15 return 0;
16 }
```



```
C:\Users\msharizal\Documents\H
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15
value of a: 16
value of a: 17
value of a: 18
value of a: 19
-----
Process exited after 0.03698
```

The counter can be represented also by a variable letter, like **a** in this case and initialized with non-zero number (which is 10).

Class Activity 2

Loops



while & do while

Video

Quiz

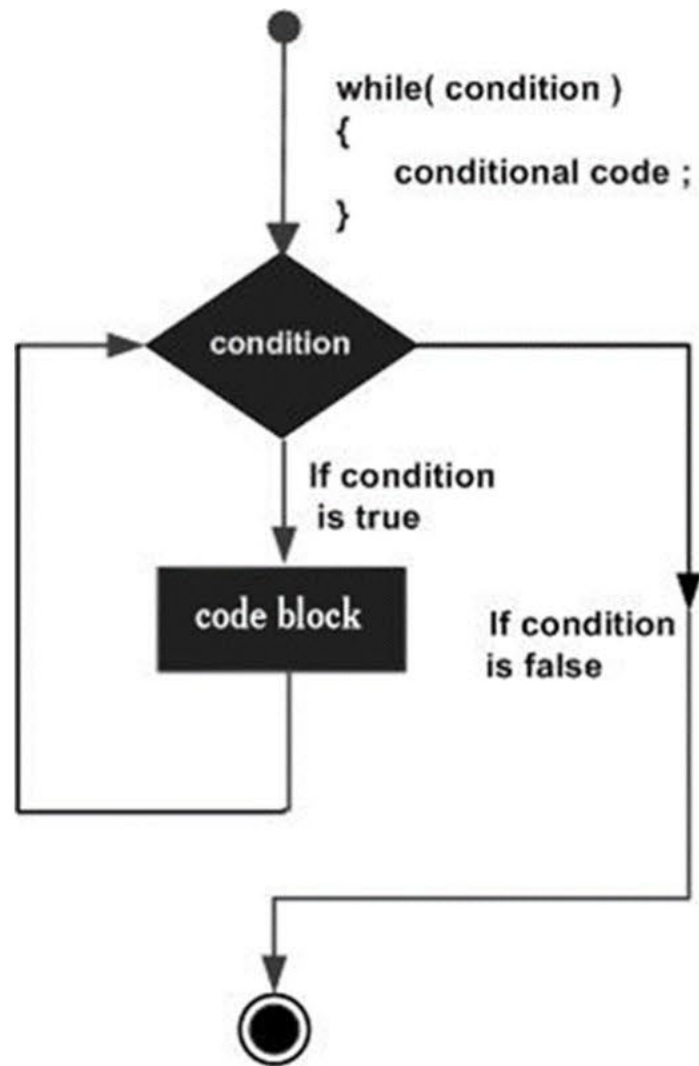
For do-while loop, at least one output will be printed out even for a FALSE case.

- ☐ TRUE
- ☐ FALSE

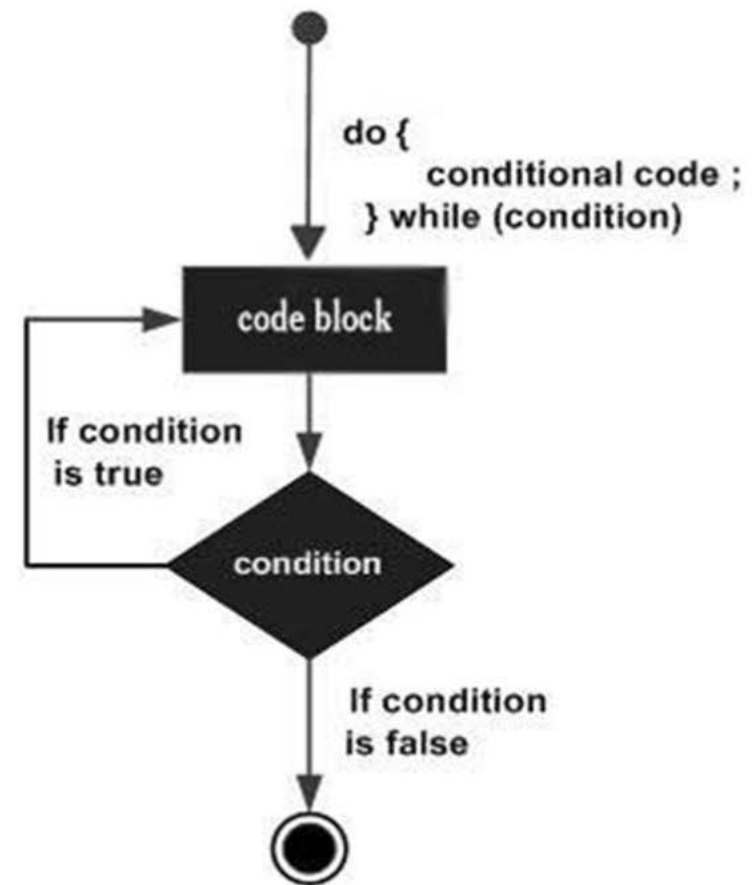


Summary

while loop



do-while loop



Higher
Colleges of
Technology



كليات
التقنية
العليا

Thank You



800 MyHCT (800 69428)



www.hct.ac.ae



HCT_UAE |



hctuae