

## CHAPTER 2.2

# CONTROL STRUCTURES (ITERATION)

Dr. Shady Yehia Elmashad



# Outline

- 1. C++ Iterative Constructs**
- 2. The for Repetition Structure**
- 3. Examples Using the for Structure**
- 4. The while Repetition Structure**
- 5. Examples Using the while Structure**
- 6. The do/while Repetition Structure**
- 7. The break and continue Statements**
- 8. Nested control structures**

# 1. C++ Iterative Constructs

- There are three constructs:
  - while statement
  - for statement
  - do-while statement

## 2. The for Repetition Structure

The general format when using **for** loops is

```
for ( initialization;  
    LoopContinuationTest; increment )  
    statement
```

Example:

```
for( int counter = 1; counter <= 10; counter++ )  
    cout << counter << endl;
```

➤ Prints the integers from one to ten

No  
semicolon  
after last  
statement

## 2. The for Repetition Structure

- Syntax

```
for (ForInit ; ForExpression ; PostExpression)  
    Action
```

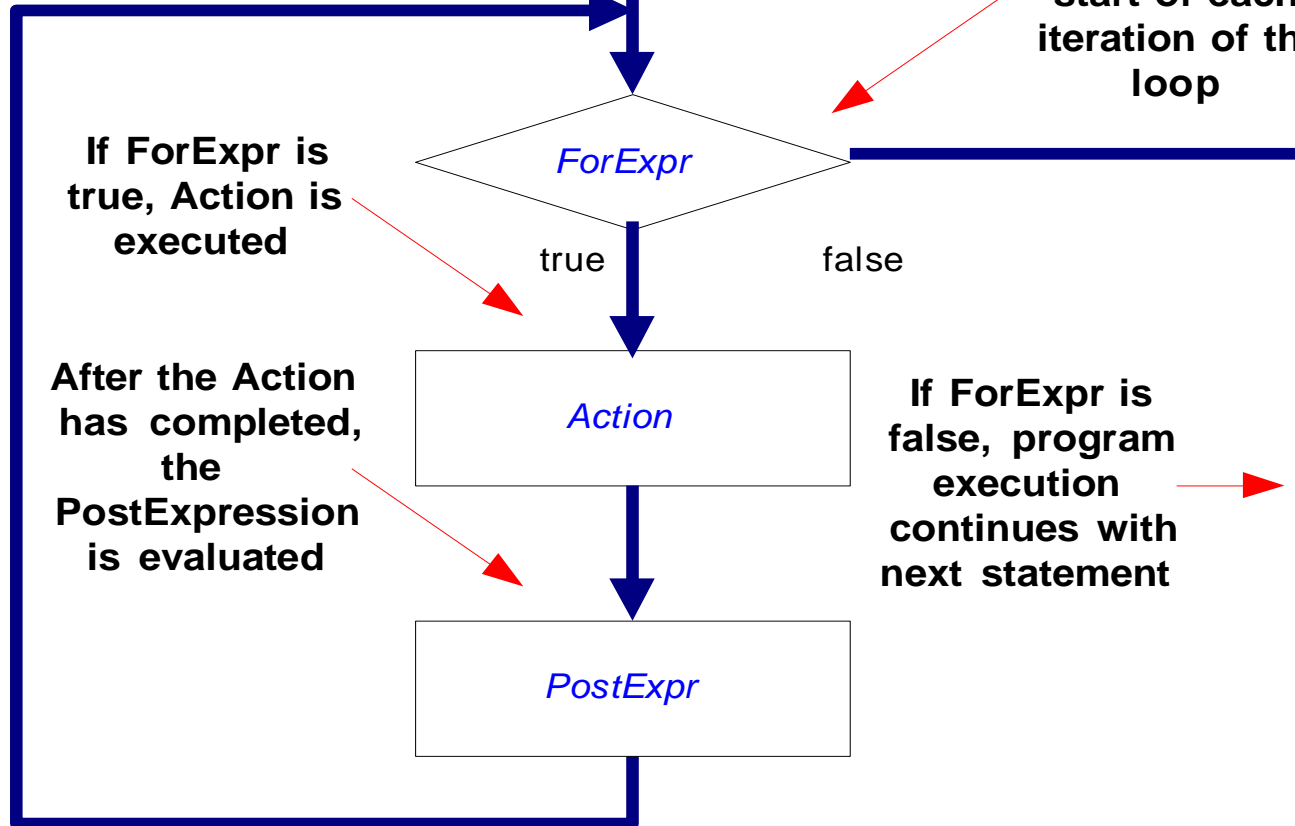
- Example

```
for (int i = 0; i < 3; ++i) {  
    cout << "i is " << i << endl;  
}
```

Evaluated once  
at the beginning  
of the for  
statements's  
execution



The ForExpr is  
evaluated at the  
start of each  
iteration of the  
loop



If ForExpr is  
true, Action is  
executed

true

false

After the Action  
has completed,  
the  
PostExpression  
is evaluated



If ForExpr is  
false, program  
execution  
continues with  
next statement



After evaluating the  
PostExpression, the next  
iteration of the loop starts

## 2. The for Repetition Structure

- **For** loops can usually be rewritten as **while** loops:

```
initialization;  
while ( loopContinuationTest) {  
    statement  
    increment;  
}
```

- Initialization and increment as comma-separated lists

```
for (int i = 0, j = 0; j + i <= 10; j++, i++)  
    cout << j + i << endl;
```

# 3. Examples Using the for Structure

## Sum the numbers from 0 to 10

```
#include <iostream.h>
void main ( )
{
int sum = 0 ;
    for ( int i = 0; i <= 10; i++ )
    {
        sum = sum + i ;
    }
cout << " Summation = " << sum ;
}
```

Summation =



# 3. Examples Using the for Structure

## Sum the even numbers from 0 to 100

```
#include <iostream.h>
void main ( )
{
int sum = 0 ;
    for ( int i = 0; i <= 100; i+=2 )
    {
        sum = sum + i ;
    }
cout << " Summation = " << sum ;
}
```

Summation =



# 3. Examples Using the for Structure

## Sum the odd numbers from 0 to 100

```
#include <iostream.h>
void main ( )
{
int sum = 0 ;
    for ( int i = 1; i <= 100; i+=2 )
    {
        sum = sum + i ;
    }
cout << " Summation = " << sum ;
}
```

Summation =

# 3. Examples Using the `for` Structure

## Printing characters depending on user entry

```
#include <iostream.h>
void main ( )
{
    int n ; char ch;
    cout << " Please enter the character: " ;
    cin >> ch ;
    cout << " Please enter the number of
    repetition: " ;
    cin >> n ;
    for ( int i = 0; i < n ; i++ )
        cout << ch;
}
```

## 4. The while Repetition Structure

Logical expression that determines whether the action is to be executed

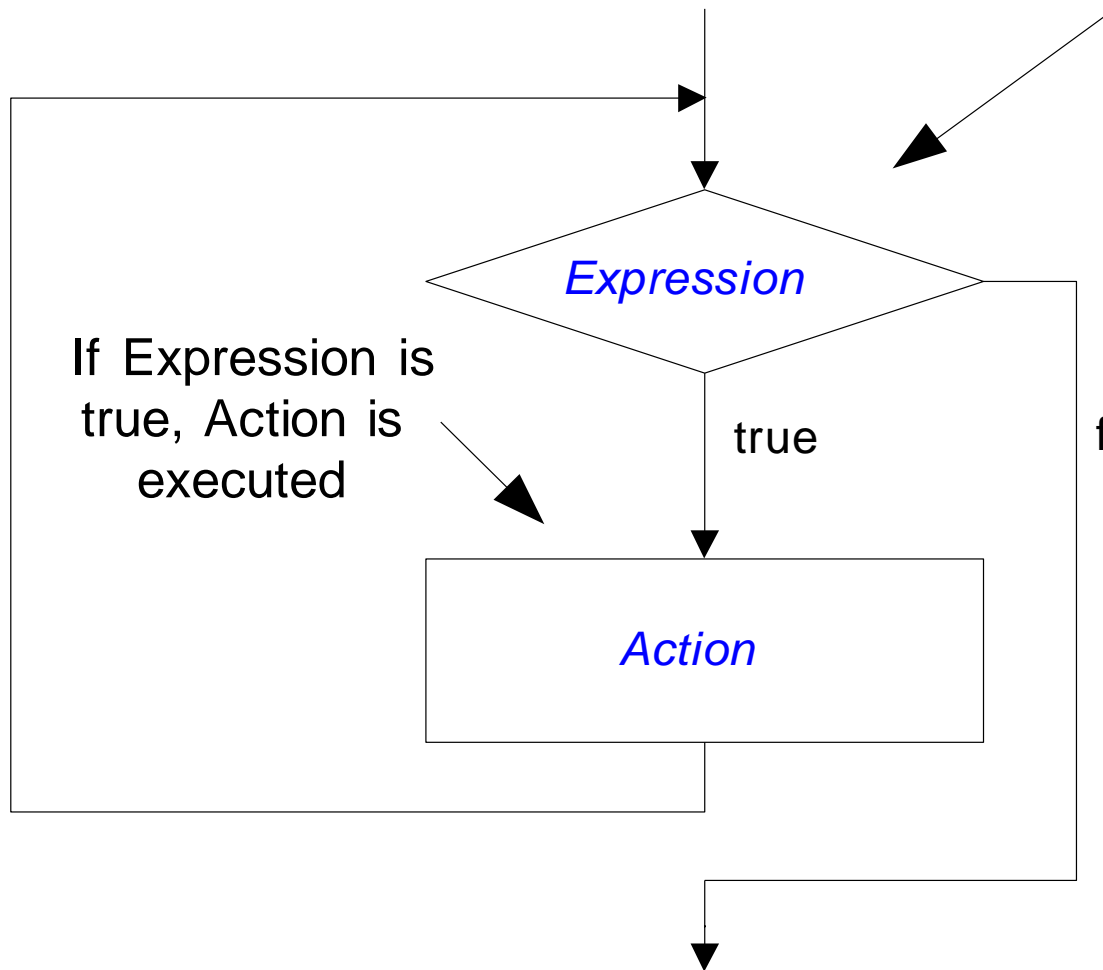
Action to be iteratively performed until logical expression is false

**while** ( *Expression*) *Action*

# 4. The while Repetition Structure

## While Semantics

Expression is evaluated at the start of each iteration of the loop



If Expression is true, Action is executed

false

If Expression is false, program execution continues with next statement

## 4. The while Repetition Structure

- Repetition structure

- Programmer specifies an action to be repeated while some condition remains true

- Psuedocode

- while there are more items on my shopping list*

- Purchase next item and cross it off my list*

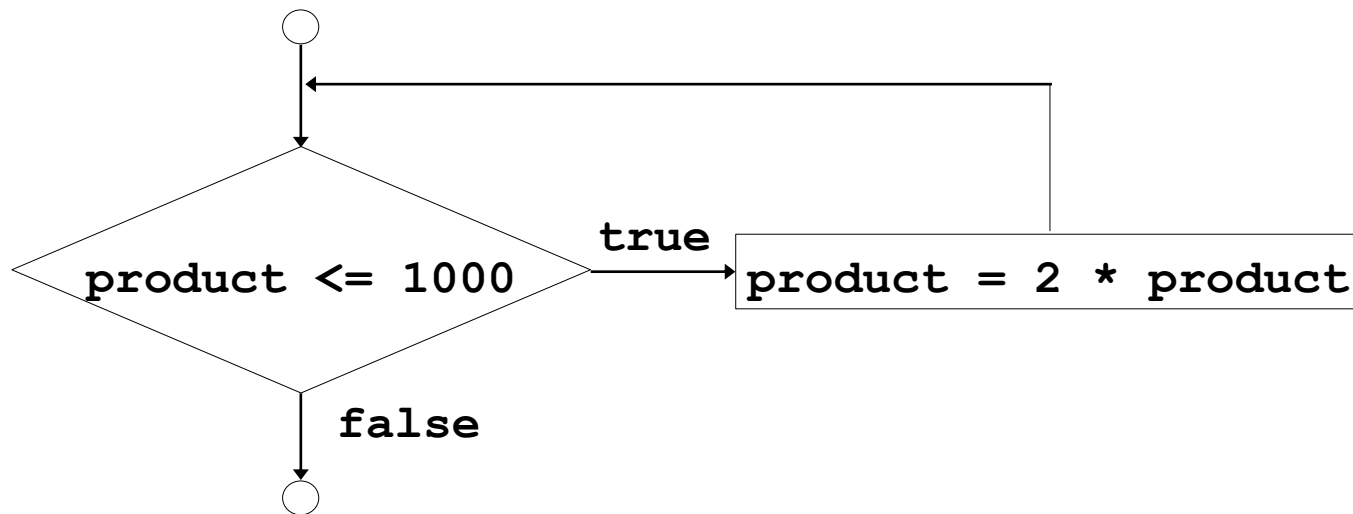
- **while** loop repeated until condition becomes false.

- Example

```
int product = 2;  
while ( product <= 1000 )  
    product = 2 * product;
```

# 4. The while Repetition Structure

- Flowchart of **while** loop



# 5. Examples Using the `while` Structure

## Printing characters depending on user entry

```
#include <iostream.h>
void main ( )
{
int n, i = 0 ; char ch;
cout << " Please enter the character: " ;
cin >> ch ;
cout << " Please enter the number of
repetition: " ;
cin >> n ;
    while ( i < n ) {
        cout << ch ;
        i ++ ;
    }
}
```



## 5. Examples Using the `while` Structure

### The summation of the numbers squared from 0 to 10

```
#include <iostream.h>
void main ( )
{
int sq_sum = 0, x = 0, y ;
    while ( x <= 10 ) {
        y = x * x ;
        sq_sum = sq_sum + y ;
        x ++ ;
    }
cout << "The summation of the
numbers squared from 0 to 10 " <<
sq_sum ;
}
```

# 5. Examples Using the `while` Structure

## Factorial of a number

```
#include <iostream.h>
void main ( )
{
int n, fact = 1 ;
cout << " Please enter a number " << endl ;
cin >> n ;
    while ( n > 0 ) {
        fact = fact * n ;
        n -- ;
    }
cout << " The factorial of your number is "
<< fact ;
}
```

# 6. The do/while Repetition Structure

- The **do/while** repetition structure is similar to the **while** structure,

- Condition for repetition tested after the body of the loop is executed

- Syntax:

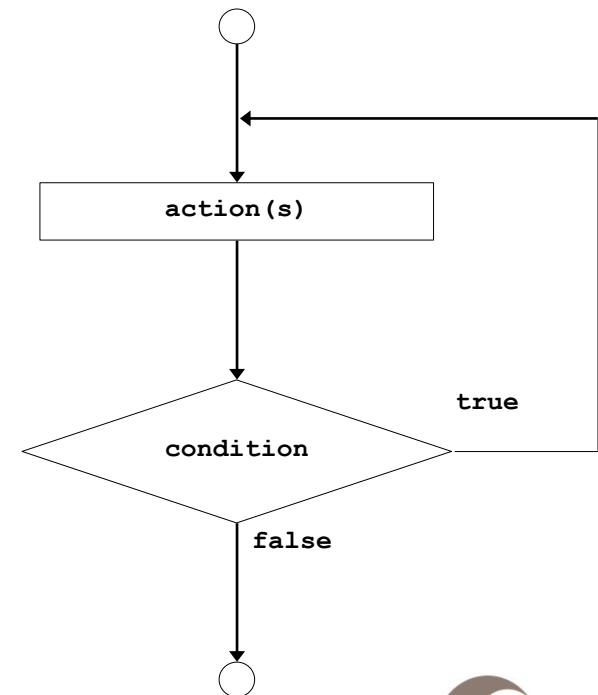
```
do {  
    statement(s)  
} while ( condition );
```

- Example (letting counter = 1):

```
do {  
    cout << counter << " ";  
} while (++counter <= 10);
```

- This prints the integers from 1 to 10

- All actions are performed at least once.



# 7. The `break` and `continue` Statements

- **Break**

- Causes immediate exit from a **while**, **for**, **do/while** or **switch** structure
- Program execution continues with the first statement after the structure
- Common uses of the **break** statement:
  - Escape early from a loop
  - Skip the remainder of a **switch** structure

# 7. The break and continue Statements

- **Continue**

- Skips the remaining statements in the body of a **while**, **for** or **do/while** structure and proceeds with the next iteration of the loop
- In **while** and **do/while**, the loop-continuation test is evaluated immediately after the **continue** statement is executed
- In the **for** structure, the increment expression is executed, then the loop-continuation test is evaluated

# 7. The break and continue Statements

```
#include <iostream.h>
Void main()
{
    int sum = 0, num;

    // Allow the user to enter up to 10 numbers
    for (int count=0; count < 10; ++count) {
        cout << "Enter a number to add, or 0 to exit: ";
        cin >> num;

        // exit loop if user enters 0
        if (num == 0)
            break;

        // otherwise add number to our sum
        sum += num;
    }
    cout << "The sum of all the numbers you entered is " << sum << "\n";
}
```

# 7. The break and continue Statements

```
#include <iostream.h>
void main ( )
{
    while (true)          // infinite loop
    {
        cout << "Enter 0 to exit or anything else to continue: ";
        int num;
        cin >> num;

        // exit loop if user enters 0
        if (num == 0)
            break;
    }

    cout << "We're out!\n";
}
```

# 7. The break and continue Statements

```
#include <iostream.h>
void main ( )
{
    for (int count=0; count < =20; ++count) {
        // if the number is divisible by 4, skip this iteration
        if ((count % 4) == 0)
            continue;

        // If the number is not divisible by 4, keep going
        cout << count << endl;
    }
}
```

- This program prints all of the numbers from 0 to 20 that aren't divisible by 4.