

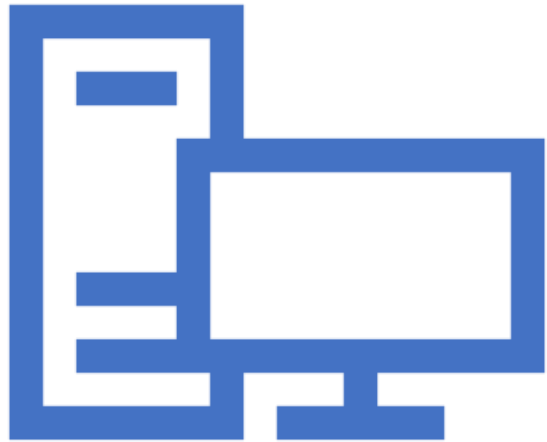
The background of the slide is a photograph of various electronic components scattered on a white surface. These include resistors with color bands, small yellow and orange capacitors, black integrated circuits, a green printed circuit board (PCB) with pins, and a green cylindrical component. A black rectangular box is overlaid on the right side of the image, containing the title and author information.

# PROGRAMMING SKILLS II

~ ANURUDDHA ABEYSINGHE

Lecture 08

Foundation Certificate in IT – Curtin Batch



# LOOP STATEMENTS

LECTURE 08

# WHY WE NEED OF A LOOP??

- There can be some situation of Same process can be execute / iterate in number of times.
  - Example – Print “Hello World..!!” in 5 times
    - Take 10 temperatures continuously in order to calculate average
    - Input students details to a system continously
- So we have to execute the same (iterate the process / loop the process) in order to get the expected results.



# ITERATIONS / LOOP STRUCTURES

- Four kinds of loops
  - while structure
  - do/while structure
  - for structure
  - foreach structure

# WHILE STATEMENT

- The *while* statement has the following syntax:

**while** is a  
reserved word

If the condition is true, the statement is executed.  
Then the condition is evaluated again.

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

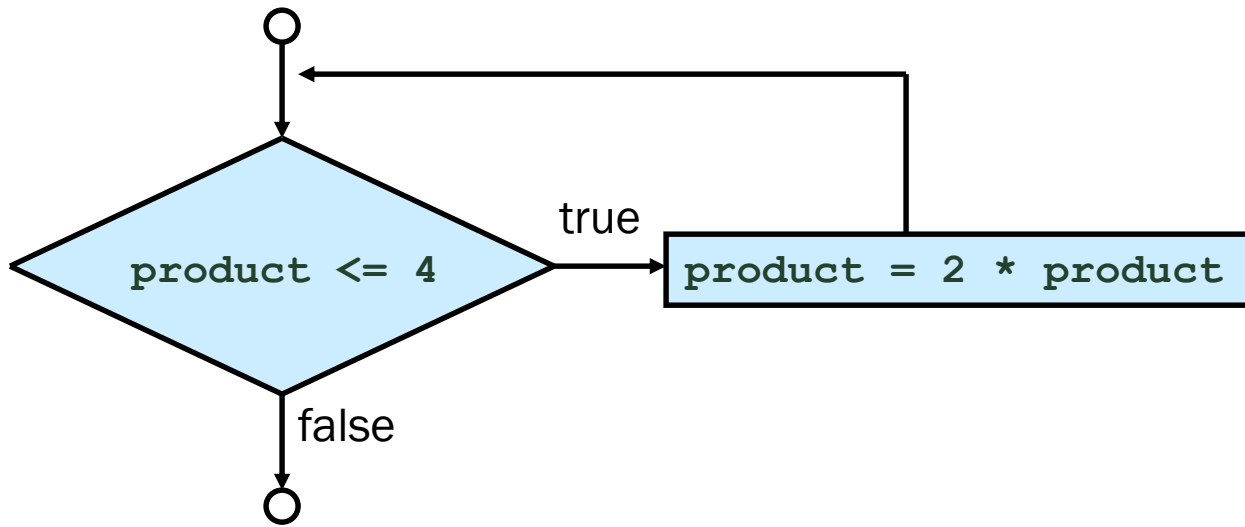
The statement (or a block of statements) is executed  
repetitively until the condition becomes false.

# WHILE STATEMENT

```
while ( <condition> )  
{  
    <code to be repeated if <condition> is true> ;  
}
```

- Repetition Structure
  - An action is to be repeated
    - Continues while <condition> is true
    - Ends when <condition> is false
  - *Contains either a line or a body of code*
  - *Use brackets to group multiple statements*

## WHILE STATEMENT (CONT'D)



```
int product;  
product = 2;  
while (product <= 4) {  
    product = 2 * product;  
}  
// beginning of the next statement
```

# THE WHILE STATEMENT

- Execute embedded statements based on Boolean value
- Evaluate Boolean expression at beginning of loop
- Execute embedded statements while Boolean value Is True

0 1 2 3 4 5 6 7 8 9

```
int i = 0;
while (i < 10) {
    Console.WriteLine(i);
    i++;
}
```



# WHILE STATEMENT

- Note that if the condition of a while statement is false initially, the statement is never executed

- Example:

```
int product;  
product = 5;  
while (product <= 4){  
    product = 2 * product;  
}  
// beginning of the next statement
```

- Therefore, the body of a while loop will execute **zero or more times**

# PROGRAMMING EXAMPLE



*Print “Hello World..!!” in 5 times*

```
int num = 1; // use increment value to get next occurrence
```

```
while(num<=5){  
    Console.WriteLine("Hello World..!!");  
  
    num = num +1; //increment by 1  
}
```

Condition

If the condition is **True** ---> Loop statements Execute

If the condition is **False** ---> Terminate the Loop

# PROGRAMMING EXAMPLE

- Write a C# program to print numbers from 1 to 100 using a while loop

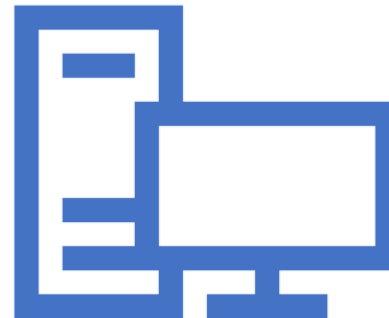
Output

1 2 3 4 5 6 7 ..... 99 100

- Modify the program to print only even numbers from 1 to 100

Output

2 4 6 8 10 12 14 ..... 98 100



# DO-WHILE STATEMENT

- Execute embedded statements based on Boolean value
- Evaluate Boolean expression at end of loop
- Even if the boolean expression is false the program execute **at least one time**
- Execute embedded statements while Boolean value Is True

```
int num = 0;  
do  
{  
    Console.WriteLine(num);  
    num = num+1;    //num++;  
}while(num<10);
```

## PROGRAMMING QUESTION

- Write a C# program to print numbers from 1 to 100 using a do-while loop
- Modify the program to print only even numbers from 1 to 100

IOIO  
IOIO

# THE FOR STATEMENT

- Place update information at the start of the loop
- Variables in a for block are scoped only within the block

The diagram illustrates the syntax of a for loop: `for (statement 1; statement 2; statement 3)`. The opening curly brace `{` is blue, and the closing curly brace `}` is also blue. The text `// code block to be executed` is written in italics between the braces. Annotations include: a purple arrow pointing to `statement 1` labeled "Loop start point"; a green arrow pointing to `statement 2` labeled "Loop End point"; a red arrow pointing to `statement 3` labeled "Increment/decrement by"; and a green arrow pointing to the semicolons separating the statements, labeled "Semicolon used to separation". The semicolons are circled in red.

```
for (statement 1; statement 2; statement 3)
{
    // code block to be executed
}
```

Example: Print 1 – 10

```
for (int i =1 ; i<=10 ; i = i+1) {  
  
    Console.WriteLine(i);  
  
}
```

# PROGRAMMING QUESTION

- Write a C# program to print numbers from 1 to 100 using a for loop
- Modify the program to print only even numbers from 1 to 100

# THE FOREACH STATEMENT

- The foreach Statement Choose the type and name of the iteration variable
- Execute embedded statements for each element of the collection class

```
string[] cars = {"Volvo", "BMW", "Ford", "Mazda"};

foreach (string i in cars)
{
    Console.WriteLine(i);
}
```



# THE **BREAK** AND **CONTINUE** STATEMENTS

- The break statement *jumps out of an iteration / Exit in iteration*
- The continue statement *jumps to the next iteration / Move to next iteration occurrence*

```
int i = 0;
while (i < 10) {
    Console.Write(i);
    i++;

    if (i == 2)
        continue;
    else if (i == 8)
        break;
}
```

Output:

0 1 2 3 4 5 6 7

# INFINITE LOOPS



The body of a while loop must eventually make the condition false



If not, it is an *infinite loop*, which will execute until the user interrupts the program



This is a common type of logical error



You should always double check to ensure that your loops will terminate normally

---

foreach  
structure

while  
structure

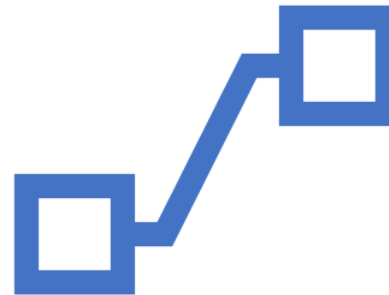
for  
structure

do/while  
structure

**LET'S  
SUMMARIZE**

## RECAP: LOOP STATEMENTS

- *Loop statements* allow us to execute a statement multiple times
- They are often simply referred to as *loops*
- Like conditional statements, they are controlled by boolean expressions
- C# has four kinds of loop statements: the *while loop*, the *do loop*, the *for loop*, and the *foreach loop*
- The programmer must choose the right kind of loop for the situation



**THANK YOU**

