Rajarata University of Sri Lanka

COM 1407 Computer Programming

LESSON 05 - FLOW CONTROL STRUCTURES II

LOOP CONTROLLING STRUCTURES/ ITERATION

BY : PIYUMI HERATH

Objectives

At the end of this lesson, the students should be able to,

- Understand the basic loop control structures used in programming.
- Explore how to construct and use count-controlled repetition structures.
- Predict the behavior of loop structures.
- Identify the importance of jump statements
- Apply the learned flow control structures in writing programs

ITERATIVE/LOOP STATEMENTS

- while
- for
- do while

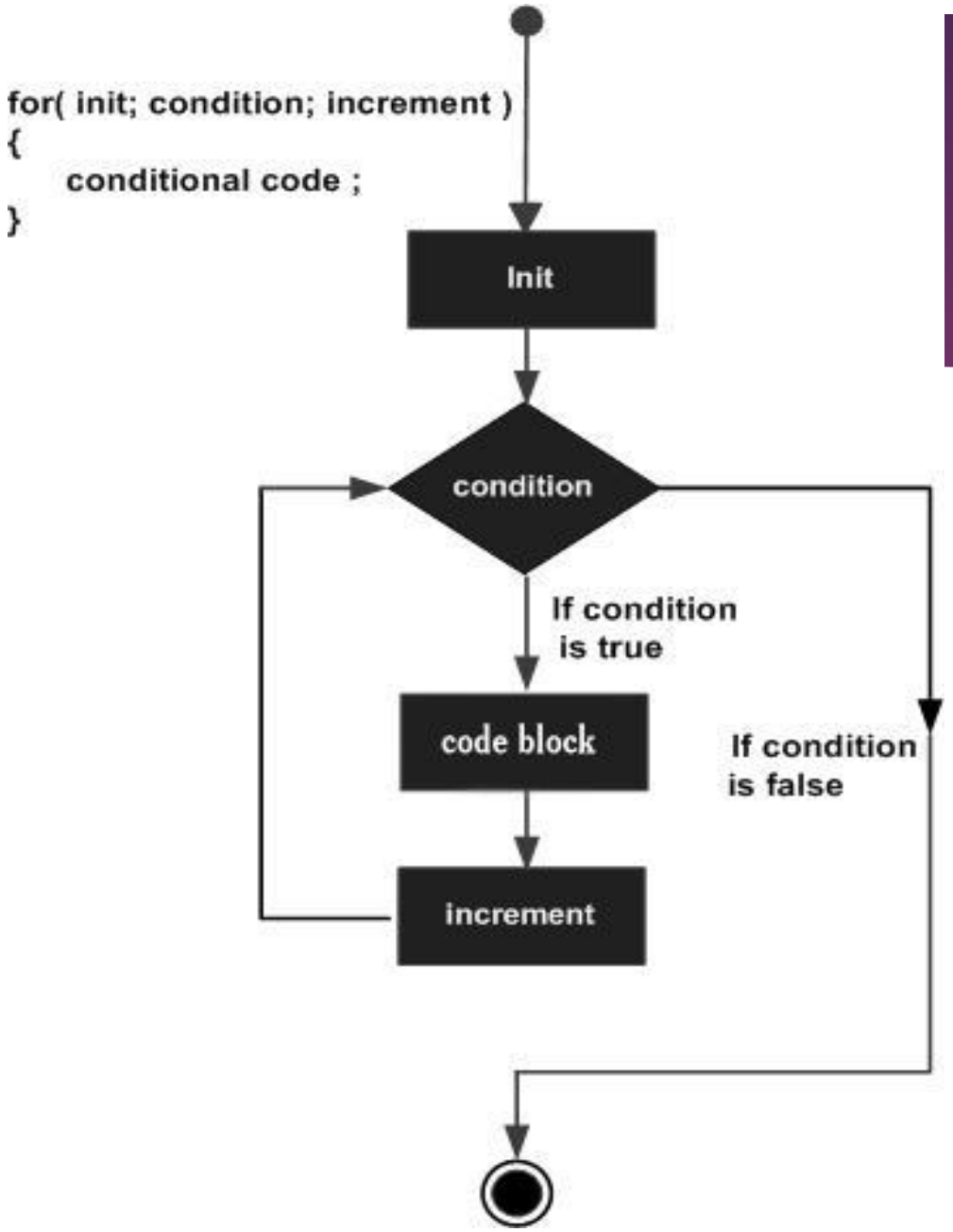
- These statements execute a set of statements over and over again, either a specified number of times or until a particular condition is being satisfied

For loop

- General form of a for loop :

```
for (<Initialization statement>; <Boolean Expression>; <Increment statement>)
{
        <statement 1>;
        <statement 2>;
        …
}
```

- First <Initialization statement> is executed.
- Next <Boolean Expression>is evaluated. If <Boolean Expression>is true, <statement 1>, <statement 2> etc. and <Increment statement> are executed.
- The <Boolean Expression>is evaluated again, and the loop continues. If <Boolean Expression>is false, executing the iterative statement is concluded.



The below for loop finds the sum of the numbers from 1 to 99 .

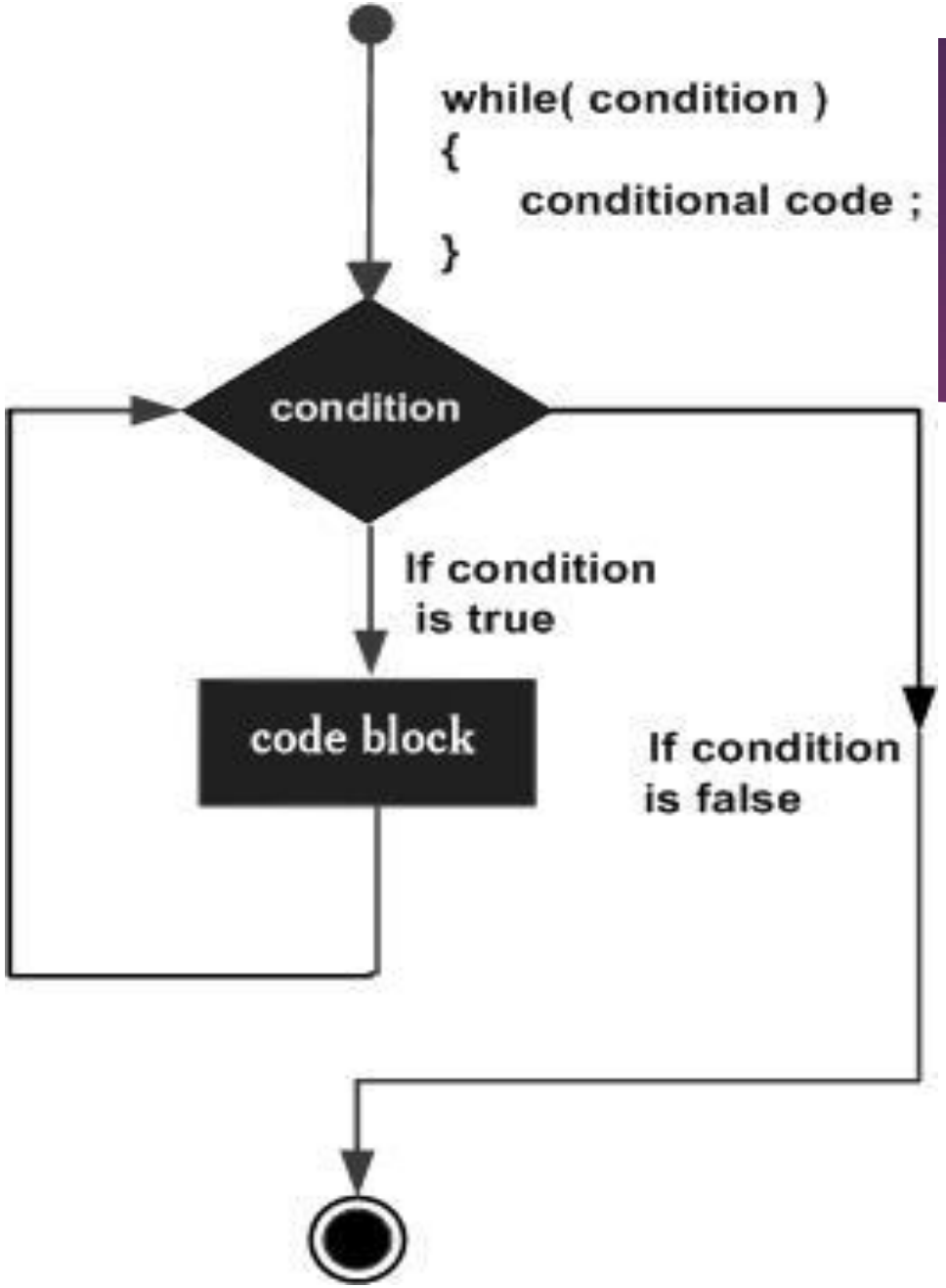
```
int i;
int sum = 0;
for (i = 0; i<100; i++)
        {
                sum = sum +i;
        }
printf(sum);
```

while loop

- General form of the while loop:

```
while (<Boolean Expression>)
{
        <statement 1>;
        <statement 2>;
        <increment statement>;
}
```

- First <Boolean Expression>is evaluated.
- If <Boolean Expression>is true, <statement 1>, <statement2> etc. are executed.
- The <Boolean Expression>is evaluated again, and the loop continues. If <Boolean Expression>is false, executing the iterative statement is concluded.



The below while loop finds the sum of the numbers from 1 to 99 .

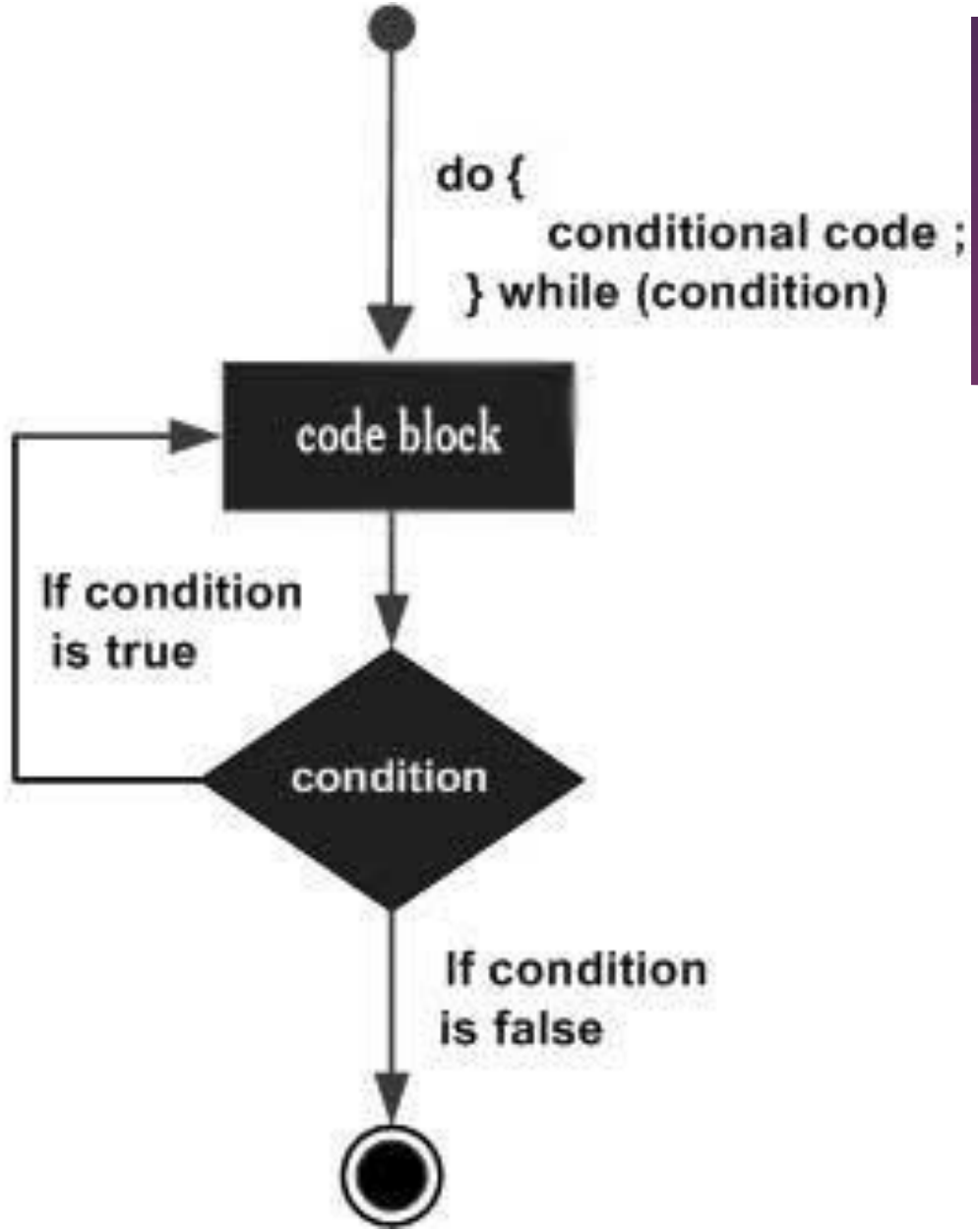
```
int sum = 0;
int i=0;
while (i<100)
{
      sum = sum +i;
      i++;
}
```

do while loop

- The general form:

```
do{
        <statement 1>;
        <statement 2>;
        <increment statement >;
        …
        } while (<Boolean Expression>);
```

- First <statement 1>, <statement 2> etc. are executed.
- Next, the <Boolean Expression>is evaluated. If <Boolean Expression>is true, <statement 1>, <statement 2> etc. are executed.
- The <Boolean Expression>is evaluated again, and the loop continues. If <Boolean Expression>is false, executing the iterative statement is concluded.



The below do-while loop finds the sum of the numbers from 1 to 99 .

```
int sum = 0;
int i=0;
do{
      sum = sum +i;
      i ++;
} while (i<100);
```

JUMP STATEMENTS

In C we have 3 types of Jump Statements.

- **break Statement**
- **continue Statement**
- **Go-to Statement**

- Jump statements move the point of execution in a program from one place to another place.

break statement

- A Break Statement jumps out of a loop and effectively bypasses the loop condition.
- The break statement is used inside loop or switch statement. When compiler finds the break statement inside a loop, compiler will abort the loop and continue to execute statements followed by loop.
- Eg: following loop finds the sum of the numbers from 1 to 50.

```
int sum = 0;
int i=0;
while (i<100){
        sum = sum +i;
        i ++;
        if (i==50)
                break;
```

Cont...

- By using this jumping statement, we can terminate the further execution of the program and transfer the control to the end of any immediate loop.
- We can specify a break jumping statements whenever we want to terminate from the loop.

continue statement

- A continue statement jumps out of the **current iteration** of a loop.
- When compiler finds the continue statement inside a loop, compiler will skip all the following statements in the loop and resume the loop.
- The following loop finds the sum of the numbers from 1 to 99 while ignoring the numbers that are multiples of 4.

```
int sum = 0, i ;
for (i = 0; i<100; i=i+1) {
    if (i%4==0)
        continue;
    sum = sum +i;
    }
```

goto statement

By using this jumping statements we can transfer the control from current location to anywhere in the program.

To do this we have to specify a **label** with *goto* and the control will transfer to the location where the label is specified.

Syntax:

```
goto label:
    .....
    .....
label:
    ......
```

Cont…

```
int main()
{
        printf("\nStatement 1.");
        printf("\nStatement 2.");

        goto last;

        printf("\nStatement 4.");
        printf("\nStatement 5.");

        last:
        printf("\nEnd of Program.");

        return 0;
}
```

Activity

What are the differences and similarities between the loop controlling statements (*for, while, do while*) that we discussed in this lesson ?

Thank You!

Next Lesson : Functions