iMSCIT

SEMESTER:I

1601102

PROBLEM SOLVING THROUGH PROGRAMMING

UNIT:III

Unit 3: Decision Control Statements and Looping Techniques

- **Decision Controls**
  - If
  - If-else
  - Nested If-else
  - Switch
- **Looping Techniques**
  - While loop
  - do-while loop
  - for loop
  - nested loop
- **Other Statements (Break, continue, GOTO, exit)**

Looping Techniques

- The looping can be defined as repeating the same process multiple times until a specific condition satisfies.
- A loop statement allows us to execute a statement or group of statements multiple times.

Advantages of Looping:

- It provides code re-usability
- Using loops, we do not need to write the same code again and again.
- Using loops, we can traverse over the elements of data structures (array or linked lists).

Why we use looping?

- The looping simplifies the complex problems into the easy ones.
- It enables us to alter the flow of the program so that instead of writing the same code again and again, we can repeat the same code for a finite number of times.
- For example, if we need to print the first 10 natural numbers then, instead of using the printf() statement 10 times, we can print inside a loop which runs up to 10 iterations.

Types Of Loop Structure

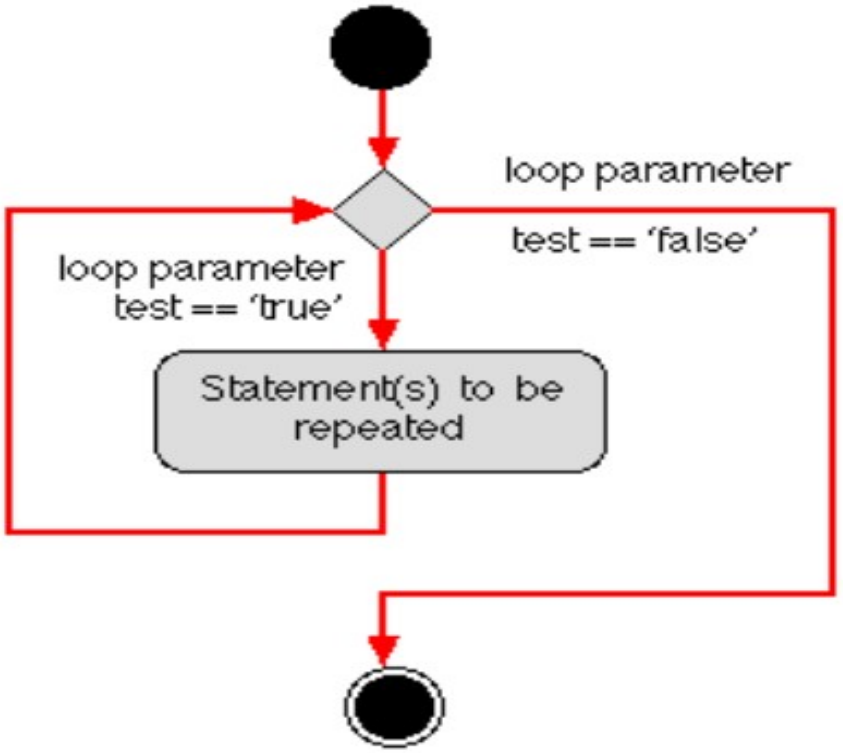
- The loop structure always has an **entry point and an exit point**.
- The entry point indicates beginning of the loop, and exit point indicates the breaking or terminating point of the loop.
- There are two types of loop structure:

1) **Pre-test or Entry Control loop**

2) **Post test or Exit Control loop**

Pre-test or Entry Control loop

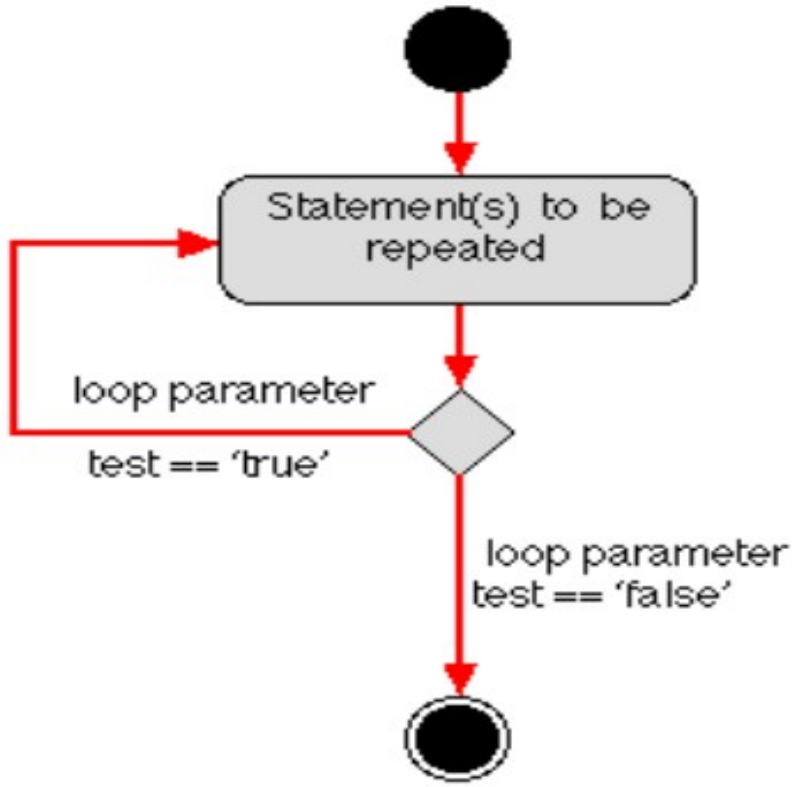
- If the control test **condition is checked at the entry point** , the loop is called a **Pre-test loop.**



(a) *Pre test loop*

Post-test or Exit Control loop

- If the control test **condition is checked at the exit point** , the loop is called a **Post-test loop.**



(b) *Post test loop*

FOR Loop

- The for loop is used in the case where we need to execute some part of the code until the given condition is satisfied.
- The for loop is also called as a per-tested loop.
- It is better to use for loop if the number of iteration is known in advance.
- The for loop in C language is used to iterate the statements or a part of the program several times.
- It is frequently used to traverse the data structures like the array and linked list.

FOR Loop

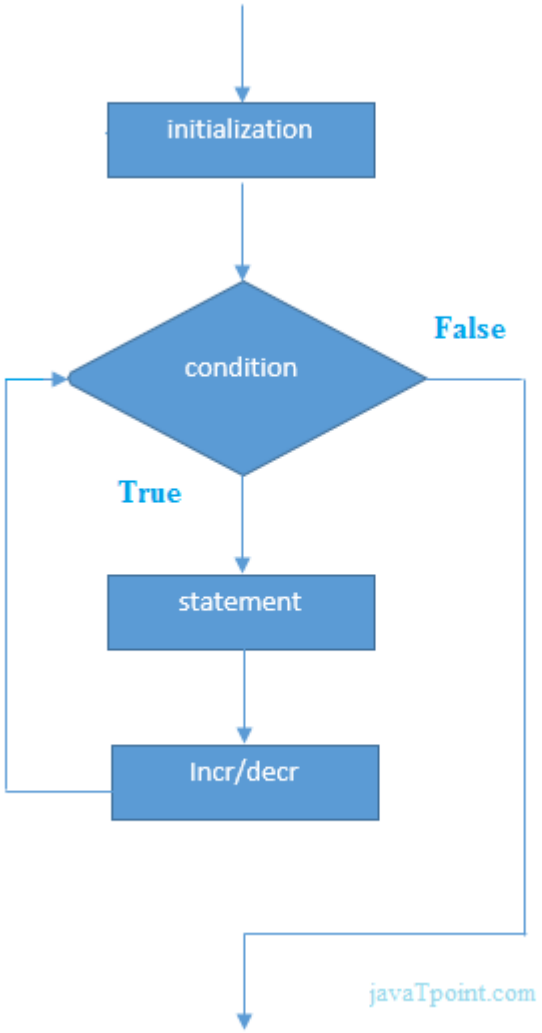
Syntax:

```
for(initialization; Condition; increment/decrem

{

//code to be executed

}
```



How for loop works?

- **The initialization step is executed first, and only once.** This step allows you to declare and initialize any loop control variables.
- **Next, the condition is evaluated**. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and the flow of control jumps to the next statement just after the 'for' loop.
- **After the body of the 'for' loop executes, the flow of control jumps back up to the increment/decrement statement.** This statement allows you to update any loop control variables. This statement can be left blank, as long as a semicolon appears after the condition.
- The condition is now evaluated again. If it is true, the loop executes and the process repeats itself (body of loop, then increment step, and then again condition).
- After the condition becomes false, the 'for' loop terminates.

How for loop works?

Properties of initialization:

- The expression represents the initialization of the loop variable.
- We can initialize more than one variable in it.
- Initialization is optional.
- In C, we can not declare the variables in initialization However, It can be an exception in some compilers.
- **Note: To make a for loop infinite, we need not give any expression in the syntax. Instead of that, we need to provide two semicolons to validate the syntax of the for loop. This will work as an infinite for loop.**

```
For( ; ; )
```

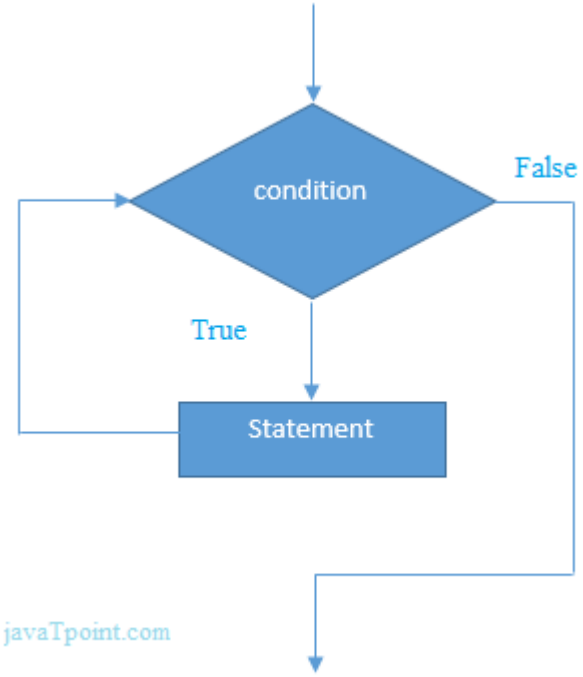
While Loop

- **The while loop in c is to be used in the scenario where we don't know the number of iterations in advance.**
- The block of statements is executed in the while loop until the condition specified in the while loop is satisfied. **It is also called a pre-tested loop.**
- In general, a while loop allows a part of the code to be executed multiple times depending upon a given Boolean condition.
- It can be viewed as a repeating if statement.
- **If the expression passed in while loop results in any non-zero value then the loop will run the infinite number of times.**

While Loop

- **Syntax:**

```
while(condition)
{
//code to be executed
}
```



- Here, statement(s) may be a single statement or a block of statements. The condition may be any expression, and true is any nonzero value. The loop iterates while the condition is true.
- When the condition becomes false, the program control passes to the line immediately following the loop.

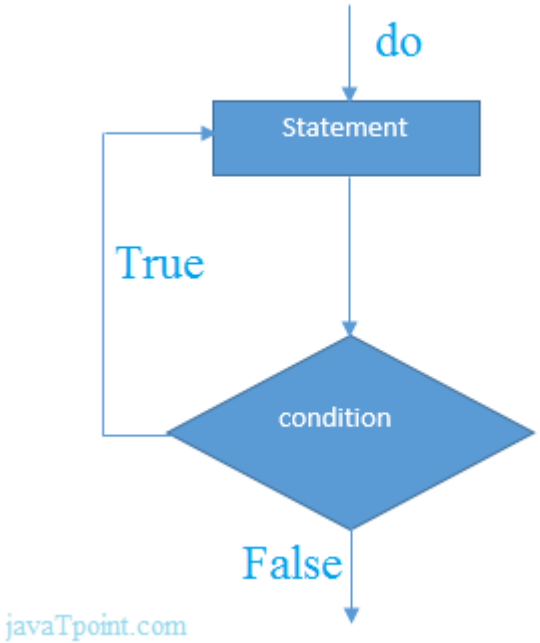
Do-While Loop

- **The do-while loop continues until a given condition satisfies. It is also called post tested loop.**
- It is used when **it is necessary to execute the loop at least once.**
- Unlike for and while loops, which test the loop condition at the top of the loop, the do...while loop in C programming **checks its condition at the bottom of the loop.**
- **The do-while loop is mostly used in menu-driven programs** where the termination condition depends upon the end user.
- The do-while loop **will run infinite times if we pass any non-zero value as the conditional expression.**

Do-While Loop

- **Syntax:**

```
do
{
//code to be executed
}while(condition);
```



- The conditional expression appears at the end of the loop, so the statement(s) in the loop executes once before the condition is tested.
- If the condition is true, the flow of control jumps back up to do, and the statement(s) in the loop executes again. This process repeats until the given condition becomes false.

Nested loops in C

- C programming allows to use one loop inside another loop.
- **Syntax:**

  The syntax for a nested for loop statement in C is as follows

```
for ( init; condition; increment ) {

   for ( init; condition; increment ) {
      statement(s);
   }
   statement(s);
}
```

Nested loops in C

- C programming allows to use one loop inside another loop.
- **Syntax:**

  The syntax for a nested for loop statement in C is as follows

```
for ( init; condition; increment ) {

   for ( init; condition; increment ) {
      statement(s);
   }
   statement(s);
}
```

Nested loops in C

The syntax for a nested while loop statement in C is as follows −

```
while(condition) {

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

Nested loops in C

The syntax for a nested do-while loop statement in C is as follows −

```
do {
   statement(s);

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

}while( condition );
```

break Statement

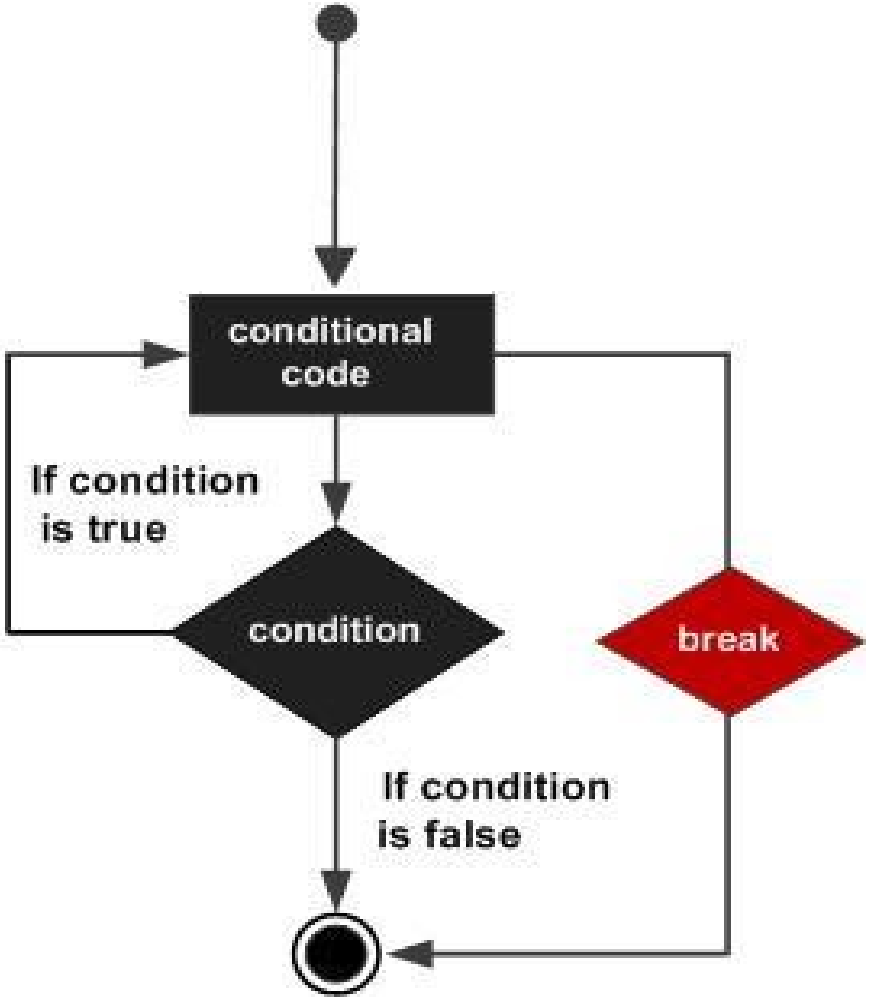
The break statement in C programming has the following two usages −

- When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop.
- It can be used to terminate a case in the switch statement
- If you are using nested loops, the break statement will stop the execution of the innermost loop and start executing the next line of code after the block.

break Statement

Syntax:

```
//loop or switch case
break;
```



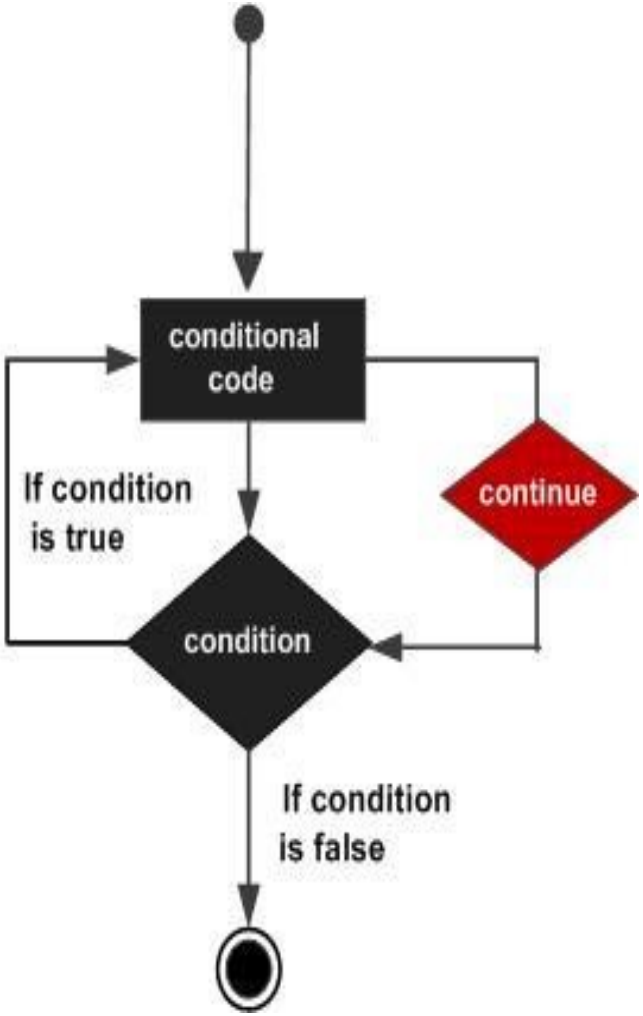
continue Statement

- **The continue statement in C language is used to bring the program control to the beginning of the loop.**
- **The continue statement skips some lines of code inside the loop and continues with the next iteration.** It is mainly used for a condition so that we can skip some code for a particular condition.
- The continue statement in C programming works somewhat like the break statement. **Instead of forcing termination, it forces the next iteration of the loop to take place, skipping any code in between.**
- For the **for loop, continue statement causes the conditional test and increment portions of the loop to execute**.
- For the **while and do...while loops, continue statement causes the program control to pass to the conditional tests.**

continue Statement

- **Syntax:**

```
//loop statements
    continue;
//some lines of the code which is to be skipped
```



goto Statement

- The goto statement is **known as jump statement in C.**
- As the name suggests, **goto is used to transfer the program control to a predefined label.**
- The goto statement can be **used to repeat some part of the code for a particular condition.**

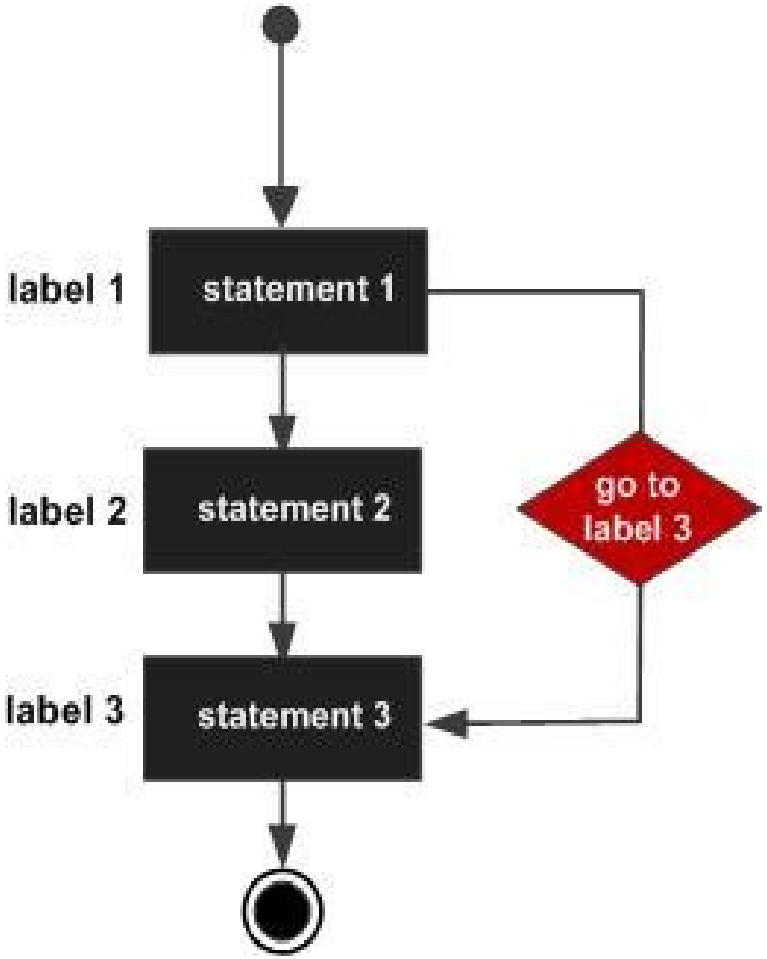
goto Statement

- **Syntax:**

label:

//some part of the code;

goto label;



THANK YOU