Decision Making and Looping

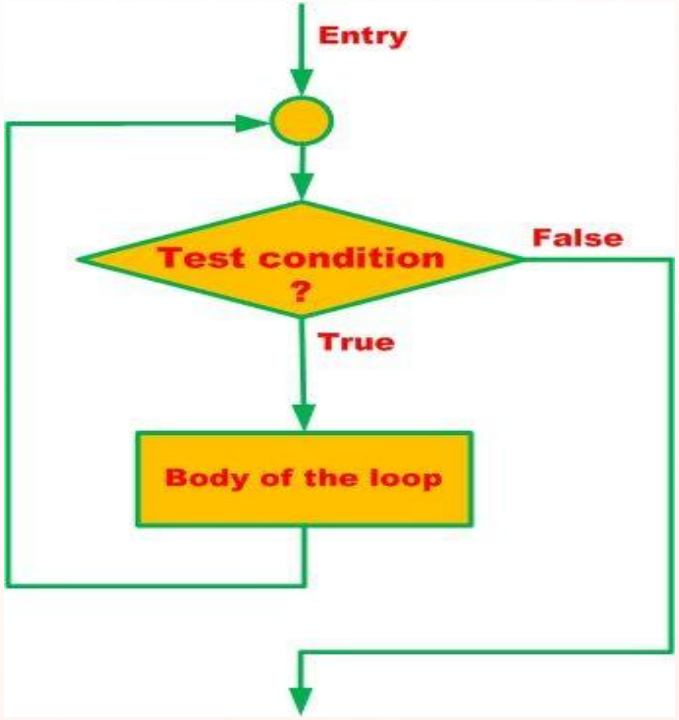


by SAHIL

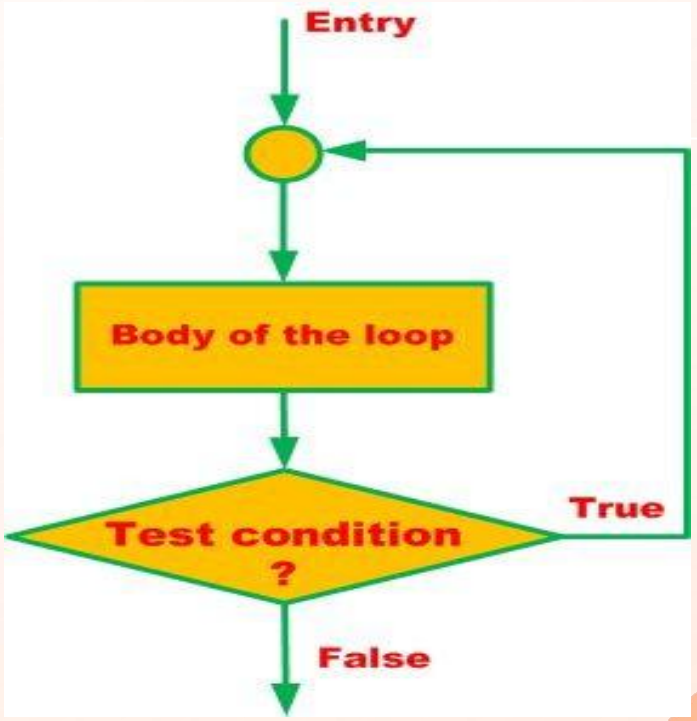
INTRODUCTION

1. *In a C program sometimes it is required to repeat a set of statements until a certain condition is met.*
2. *The repetition of set of statements depending upon result of a test condition is called as looping.*
3. *A loop essentially consists of two segments, one is body of the loop and other one is control statement.*
4. *The control statement tests certain condition and depending upon the result, directs the control of execution inside the body of the loop or to the next statement which would be present after the loop.*

5. *Depending on the position of the control statement loops are classified as the entry-controlled loop and exit-control loop.*



Entry-controlled loop



Exit-controlled loop

6. *A looping process in general would include the following four steps.*
   a. *Setting and initialization of a condition variable.*
   b. *Execution of the statements in the loop.*
   c. *Test for a specified value of the condition variable for execution of the loop.*
   d. *Updating the condition variable.*

Loops in C

1. *while loop/statement*
2. *do while loop/statement*
3. *for loop/statement*

while loop:

1. *The while loop is an entry-controlled loop.*
2. *It takes the general form as:*

```
while (test condition)
{
    Body of the loop
}
```

3. *The test condition is evaluated first and if the condition is true, then the body of the loop is executed.*
4. *After execution of the body the test condition is once again evaluated and if it is true, the body is executed once again.*
5. *The body of the loop continues to execute until the test condition finally becomes false and the control is transferred out of the loop.*
6. *On exit the program continues with the statement immediately after the body of the loop.*

do while loop:

1. The do while loop is an exit-controlled loop.
2. It takes the general form as:

```
do
{
    Body of the loop
}
while (test condition);
```

3. The body of the loop is executed for the first time irrespective of the test condition.

4. At the end of the loop test condition with while statement is evaluated.
5. If the condition is true the body of the loop is once gain executed.
6. The body of the loop continues to execute until the test condition finally becomes *false* and the control is transferred *out of the loop*.

for loop:

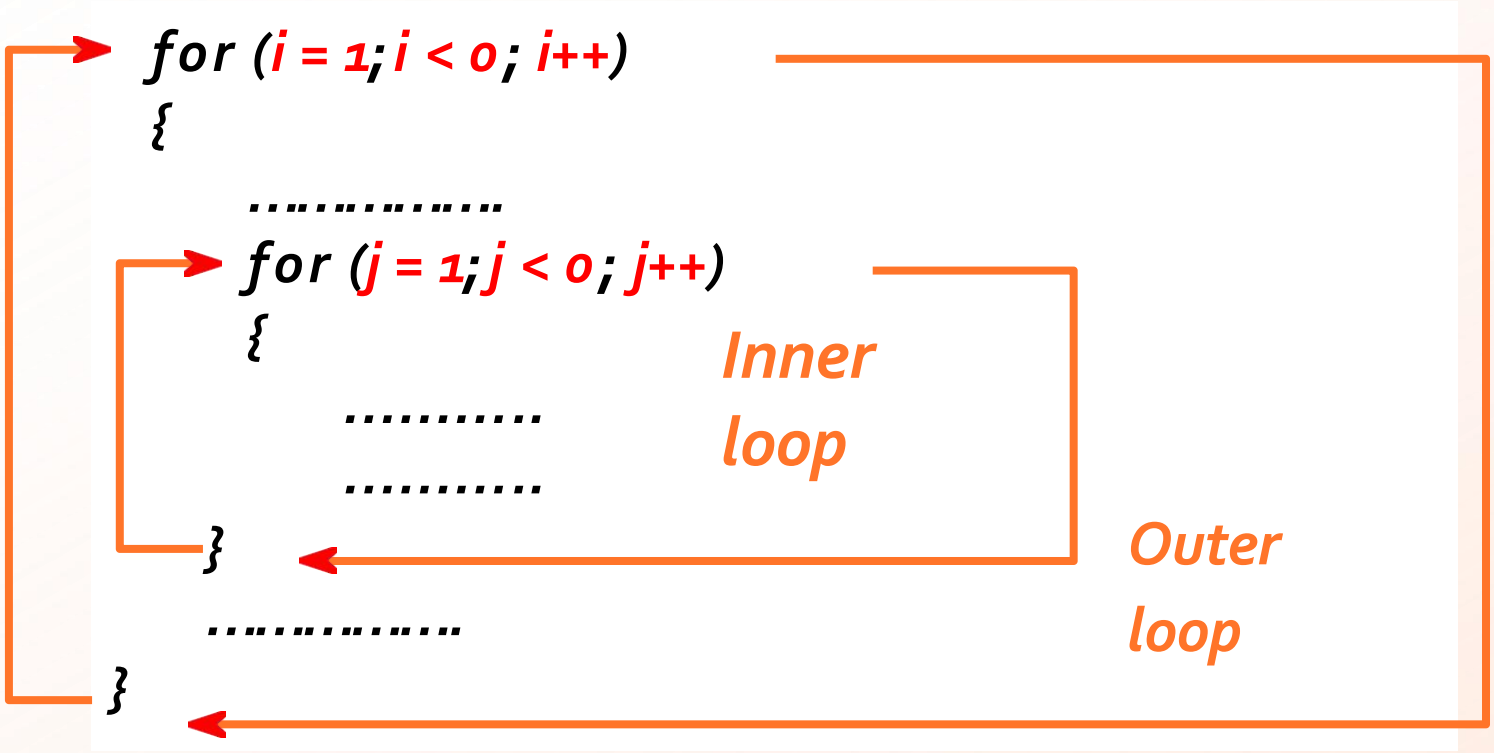
1. The for is another entry-controlled loop.
2. It takes the general form as:

```
for (initialization; test-condition; updation)
{
    Body of the loop
}
```

3. The execution of the for statement begins with initialization of loop control variable such as i = 0.
4. The value of test control variable is tested using the test-condition.

3. *If the test condition is true the body of the loop is executed and control is transferred back to the for statement.*
4. *Now the control variable is updated (incremented or decremented).*
5. *The new value of control variable is again tested and if results into true the body of the loop is once again executed.*
6. *The body of the loop continues to execute until the test condition finally becomes false and the control is transferred out of the loop.*

Nesting of *for* loops:



break statement in c

1. *It is used for early exit from the loop on the occurance certain condition.*
2. *When break statement is encountered by the compiler, the loop is immediately terminated irrespective of the test condition specified for the loop and the control is transferred to the statement following the loop.*
3. *When the loops are nested then the break statement would terminate the loop containing it only.*

```
while (.....)
{
    .....
    .....
    if (condition)
    break;
Loop .....
    .....
exit
}
    .....
```

```
do
{
    .....
    .....
    if (condition)
    break;
Loop .....
    .....
exit
}while (.....);
    .....
```

```
for (.....)
{
    .....
    .....
    if (condition)
    break;
    .....
    .....
}

.....
```

Loop exit

```
for (.....)
{
    .....
    for (.....)
    {
        if (condition)
        break;
        .....
    }
    .....
}

.....
```

inner Loop exit

continue statement in c

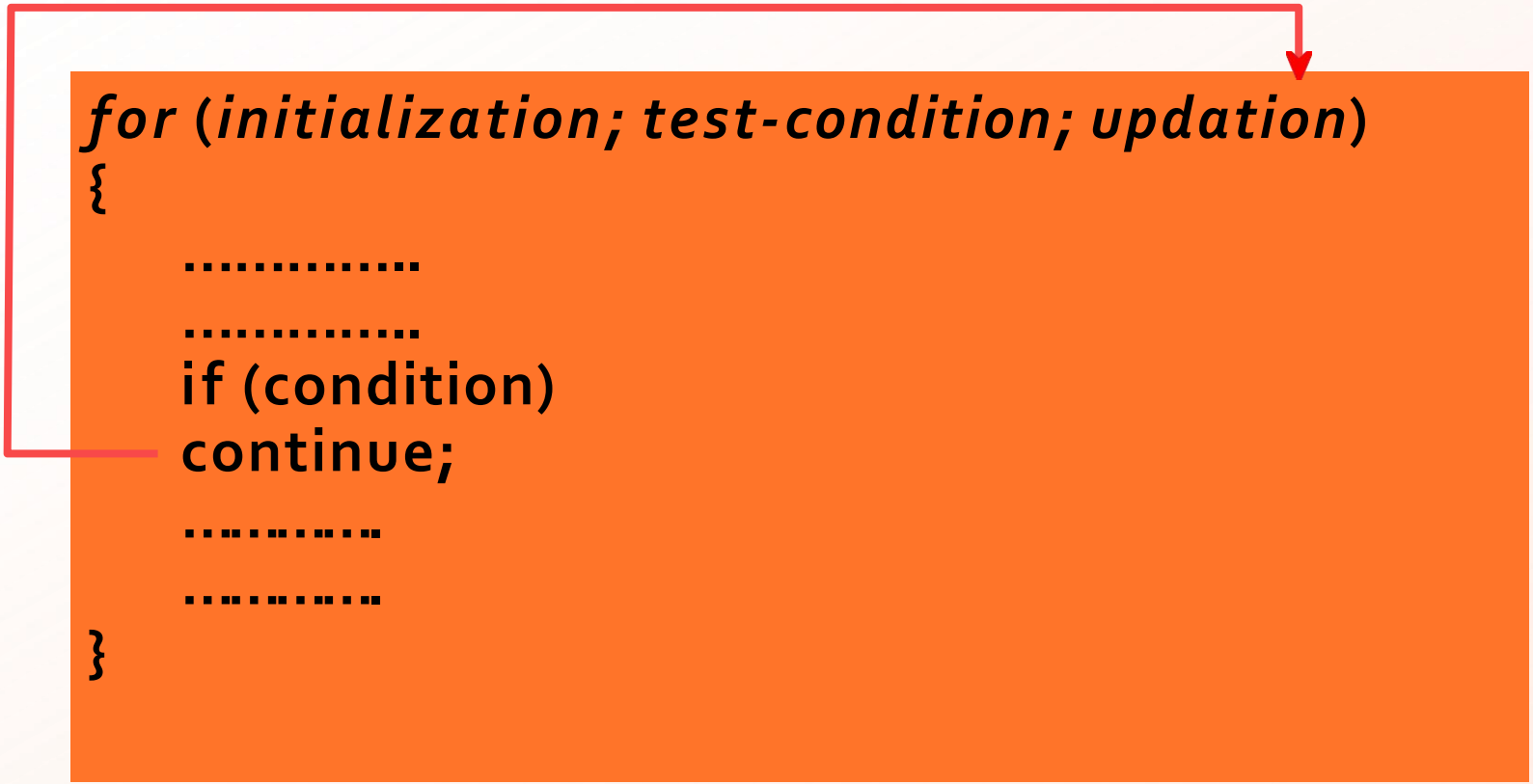
1. *It is used to skip a part of a loop on the occurrence of certain condition .*
2. *When continue statement is encountered by the compiler in a particular iteration of a loop, the statements following it are not executed i.e. skipped and the compiler continues with the next loop iteration.*
3. *In while and do while loops the control is transferred to the test condition whereas in for loop the control is transferred to the updation section.*

```
while (.....)
{
    .....
    .....
    if (condition)
    continue;
    .....
    .....

}
```

```
do
{
    .....
    .....
    if (condition)
    continue;
    .....
    .....

}while (.....);
```


for (initialization; test-condition; updation)
{
.....
.....
if (condition)
continue;
.....
.....
}

Thank you



by SAHIL