Looping Control Structure (for, while, do…while)

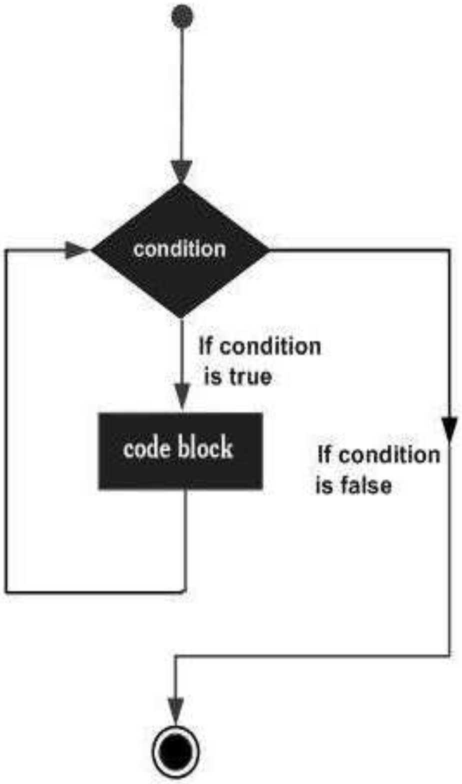
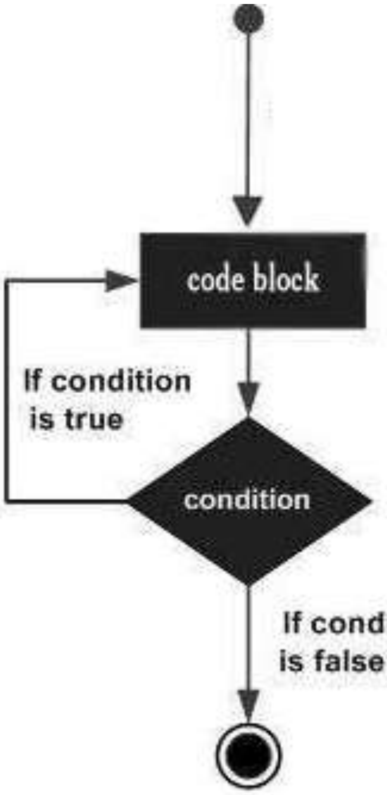
You may encounter situations when a block of code needs to be executed several number of times. In general, statements are executed sequentially: The first statement in a function is executed first, followed by the second, and so on till the condition is satisfied. So when this type of code is needed we will use looping structure.

1 Explain loops available in C with example. OR Explain Entry Control loop and Exit control loop in example.

- Loops are used to repeat execution of a block of code. Repetitive Execution of a program is called a looping structure.
- During looping, a set of statements are executed until some condition for termination is encountered.
- So based on condition loop is divided into two categories
  1. Entry control loop
  2. Exit control loop.
- **1. Entry control loop**: in this , condition is written first and then body of the loop or body of looping statement is executed till the condition is true and when the condition is false, loop will be terminated. So if condition is true then only entry control loop will be executed else it will not be executed for a single time.

- **2 Exit Control Loop:** in this, body of the loop or body of looping statement is written first and then condition of the loop. **So loop is executed at least once, a single time if condition is false at first time**. And for next iteration if the condition is true then again body of the loop will be executed and when the condition is false, loop will be terminated. So if condition is false for the **1st time then also exit control loop will be executed at least one time**

  Flow chart for entry control loop and Exit Control loop:

Entry Control loop	Exit Control loop
 condition If condition is true code block If condition is false	 code block If condition is true condition If condition is false

C supports three types of looping (Two are entry control loops and one is exit control loop)

1. **while loop**

- The simplest of all looping structure is while statement.
- The general format of the while statement is:
- Syntax:

```
Initialization;
while (test condition)
{
        body of the loop
        ;
        increment or
        decrement;
}
```

- Test condition is evaluated and if the condition is true then the body of the loop is executed.
- After the execution of the body, the test condition is once again evaluated and if it is true, the body is executed once again.
- This process is repeated till the test condition is true. When it becomes false, the control is transferred out of the loop.
- On exit, the program continues with the statements immediately after the body of the loop.
- **While loop is also known as <u>entry control loop</u>** because first control-statement is executed and if it is true then only body of the loop will be executed.

Example: To print first 10 positive integer numbers

```
#inlcude<stdio.h>
#inlcude<conio.h>
void main()
{
        int i;
        i = 1;                                  \\ initialization of i
        while(i<= 10)        \\ condition checking
        {
                printf("\t%d",i);               \\ statement execution
                i++;                            \\ increment of control variable
        }
        getch();
}
```

2. **For loop**

- *for* loop provides a more concise loop controlstructure.
- The general form of the for loop is:

```
for (initialization; test condition; increment/decrement)
{
        body of the loop;
}
```

- When the control enters for loop, the variables used in for loop is initialized with the starting value such as i=0, count=0. Initialization part will be executed exactly one time.
- Then it is checked with the given test condition. If the given condition is satisfied, the control enters into the body of the loop. If condition is not satisfied then it will exit the loop.
- After the completion of the execution of the loop, the control is transferred back to the increment part of the loop. The control variable is incremented using an assignment statement such as i++
- If new value of the control variable satisfies the loop condition then the body of the loop is again executed. This process goes on till the control variable fails to satisfy the condition.

- **For loop is also <u>entry control loop</u>** because first control-statement is executed and if it is true then only body of the loop will be executed.

  Example display 1 to 10 number using for loop:

```
#inlcude<stdio.h>
#inlcude<conio.h>
void main()
{
        int i;
        for(i = 1;i<=10;i++)                \\ initialization of I , condition and increment /
        decrement of i
        {
                printf("\t%d",i);           \\ statement execution
                i++;                        \\ increment of control variable
        }
        getch();
}
```

3. do…while loop

- In contrast to while loop and for loop, the body of the do…while loop is executed first and then the loop condition is checked.
- **The body of the loop is executed at least once** because do…while loop tests condition at the bottom of the loop after executing the body.
- **do…while loop is also known as <u>exit control loop</u>** because first body statements are executed and then control-statement is executed, thus condition checking happens at exit point.
- The general format of the do…while statement is:

```
initi
aliz
atio
n;

do
{
    statement;
    increment or decrement;
}
while(test-condition);
```

Example: To print first 10 positive integer numbers

```
#inlcude<stdio.h>
#inlcude<conio.h>

void main()
{
        int i;
        i = 1;                              \\ initialization
        of i do
        {
                printf("\t%d",i);           \\ statement execution
                i++;                        \\ increment of control variable
        } while(i<= 10);                     \\ condition checking
        getch();
}
```

Sum of first 10 number using for, while and do…while loop …..its looping structure…..

for Loop	while Loop	do…while Loop

for(i=1; i < = 10; i++)
{
sum = sum + i ;
}
	i=1;
	while(i<=10)
	{
	sum = sum + i; i ++;
	}
		i=1;
		do
		{
		sum = sum + i; i ++;
		}

2 State the difference between entry control loop and exit controlloop.

Entry control loop	Exit control loop
Entry control loop checks condition first and then body of the loop will be executed.	Exit control loop first executes body of the loop and checks condition at last.
Body of loop may or may not be executed at all.	Body of loop will be executed at least once because condition is checked at last.
for, while are example of entry control loop.	Do…while is example of exit control loop.

3 Explain break and continue with example.

1. Break Statement

- Sometimes it is required to quit the loop as soon as certain condition occurs.
- A *break* statement is a flow breaking statement which is used to jump out of a loop or to break the execution of the loop.
- A *break* statement provides an early exit from for, while, do…while and switch statements.
- A *break* causes the innermost enclosing loop or switch to be exited immediately.
- **break;** keyword is used to perform this task.

2 Continue..Statement

- The *continue* statement is also a flow breaking statement the same as break statement but it works in different manner.
- It can be used to skip the rest of the body of an iterative loop.
- The *continue* statement tells the compiler, “SKIP THE FOLLOWING STATEMENTS AND CONTINUE WITH THE NEXT ITERATION”.

Write a Program to enter the number from user….print sum of all even numbers entered by user with the help of break and continue….

```
#include<stdio.h>
#inlcude<conio.h>
void main()
{
    int num, sum=0;
    clrscr();
    for( ; ;)
    {
        printf(“\n\nenter the num”);
        scanf(“%d”,&num);
        if(num==999)   // value not given so assumed by user for
             break;            // break;
      else if(num%2==1)
             continue;
     else
             sum=sum+num;
   }
   printf(“sum of even numbers entered by user is %d”, sum);
   getch();
}
```