Types of Loop

There are 3 types of Loop in C language, namely:

1. while loop
2. for loop
3. do while loop

**while loop**

while loop can be addressed as an **entry control** loop. It is completed in 3 steps.

* Variable initialization.(e.g int x = 0;)
* condition(e.g while(x <= 10))
* Variable increment or decrement ( x++ or x-- or x = x + 2 )

**Ex : print numbers from 1 to 10**

#include<stdio.h>

void main( )

{

int x;

x = 1;

while(x <= 10)

{

printf("%d\t", x);

/\* below statement means, do x = x+1, increment x by 1\*/

x++;

}

}

**Do while Loop**

In some situations it is necessary to execute body of the loop before testing the condition. Such situations can be handled with the help of do-while loop. do statement evaluates the body of the loop first and at the end, the condition is checked using while statement. **It means that the body of the loop will be executed at least once, even though the starting condition inside while is initialized to be false.**

**Ex : print multiples of 5 using do while**

#include<stdio.h>

void main()

{

int a, i;

a = 5;

i = 1;

do

{

printf("%d\t", a\*i);

i++;

}

while(i <= 10);

}

**Example : while and do while**

#include <stdio.h>

int main()

{

printf("how many times you want to print\n");

int number;

scanf("%d",&number);

printf("---------\n");

// while(number >0){

// printf("%d\n",number);

// number = number - 1;

// }

do{

printf("%d\n",number);

number = number - 1;

}while(number > 0);

//for input : -2 : do while works

return 0;

}

**For loop**

for loop is used to execute a set of statements repeatedly until a particular condition is satisfied. We can say it is an **open ended loop.**. General format is,

for(initialization; condition; increment/decrement)  
{  
 statement-block;  
}

In for loop we have exactly two semicolons, one after initialization and second after the condition. In this loop we can have more than one initialization or increment/decrement, separated using comma operator. But it can have only one **condition**.

The for loop is executed as follows:

1. It first evaluates the initialization code.
2. Then it checks the condition expression.
3. If it is **true**, it executes the for-loop body.
4. Then it evaluate the increment/decrement condition and again follows from step 2.
5. When the condition expression becomes **false**, it exits the loop.

**Ex : print numbers 1 to 10**

#include<stdio.h>

void main( )

{

int x;

for(x = 1; x <= 10; x++)

{

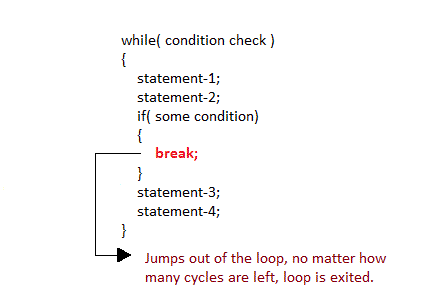
printf("%d\t", x);

}

}

**Break and Continue statements**

**Break statement**

When break statement is encountered inside a loop, the loop is immediately exited and the program continues with the statement immediately following the loop. 

**Continue Statement**

It causes the control to go directly to the test-condition and then continue the loop process. On encountering continue, cursor leave the current cycle of loop, and starts with the next cycle.

