What is Looping ?

- The looping can be defined as repeating the same process multiple times until a specific condition satisfies. It is known as iteration also.

There are three types of loops used in the C language .

Why looping?

• The looping simplifies the complex problems into the easy ones.

• It enables to alter the flow of the program so that instead of writing the same code again and again, we can execute the same code for a finite number of times.

For example, if we need to print 'AMRITA UNIVERSITY' 10-times then, instead of using the printf statement 10 times, we can use printf once inside a loop which runs up to 10 iterations.

What are the advantages of Looping?

1) It provides code reusability.

2) Using loops, we do not need to write the same code again and again.

3) Using loops, we can traverse over the elements of data structures structures (array or linked lists).

Types of C Loops

There are three types of loops in C language those are given below:

- while
- do while
- For loop

Essential **components** of a loop

- Counter
- Initialisation of the counter with initial value
- Condition to check with the optimum value of the counter
- Statement(s) to be executed by iteration
- Increment/decrement

while loop in C

- The while loop in c is to be used in the scenario where 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.
- The syntax of while loop in c language is given below:

```
 initialisation;
 while(condition)
{
block of statements to be executed ;
increment;
 }
```

Program to print table for the given number using while loop in C

```
#include main()

{ int i=1,number,b;

 printf("Enter a number: ");

scanf("%d",&number);

while(i<=10)

 {

  b=number*i;

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

   i=i++;

 }

}
```

Output

```
Enter a number: 5
 5
10
15
20
25
30
35
40
45
50
```

do-while loop in C

- The do-while loop continues until a given condition satisfies.
- It is also post tested loop.
- It is used when called it is necessary to execute the loop at least once (mostly menu driven programs).
- The syntax of The syntax of do-while loop in c language is given below:

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

Example: Program to print multipliction table of a number

```
#include <stdio.h>
Int  main()
{   int number, i,m;                          /* declaration */
 printf( "Enter the number\n" );
 scanf( "%d", &number);          /* read a number */
  i=1;
 Do
   {                                          /* Starting of loop     */
     m=number*i;                      /* calculation of m*/
     printf(“%d, ”, m);                  /*printing m */
    }while(i<=10);                      /* condition testing*/
 }
```

for loop in C

- 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.
- The syntax of for loop in c language is given below: for loop in C

```
for(Expression1; Expression2; Expression3)
  {
      codes to be executed;
  }
```

Expression 1 (Optional)

- Represents the initialization of the loop variable.
- More than one variable can be initialised.

Expression 2

- Expression 2 is a conditional expression. It checks for a specific condition to be satisfied. If it is not, the loop is terminated.
- Expression 2 can have more than one condition. However, the loop will iterate until the last condition becomes false. Other conditions will be treated as statements.

Expression 3

- Expression 3 is increment or decrement to update the value of the loop variable

Program to print natural numbers 1to15

```
#include main()
 {
 int i;
 for(i=1;i<=15;i=i+1)
   {
     printf("%d, ", i);
   }
 }
```

```
output  ---1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,
```

Loop Control Statements-: **Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed.**

break statement -Terminates the loop or switch statement and transfers execution to the statement immediately following the loop or switch.

continue statement -Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating.

goto statement- Transfers control to the labeled statement. Though it is not advised to use goto statement in your program.

The Infinite Loop: A loop becomes infinite loop if a condition never becomes false. The for loop is traditionally used for this purpose. Since none of the three expressions that form the for loop are required, you can make an endless loop by leaving the conditional expression empty.

```
#include<stdio.h>
 int main ()
{
   for( ; ; )
   {
      printf("This loop will run forever.\n");
   } return 0;
 }
```