**Rajarata University of Sri Lanka**

**Faculty of Applied Sciences**

**COM 1407 – Computer Programming**

**Practical – Loops in C**

---

**Outline**

- For Loop
- While Loop
- Do – While Loop
- break and continue

**Outcome**

- Be Familiar with loops in C
- Get knowledge about general structures of Loops
- Write C programs using loops

---

# 1 for Loop

A **for** loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times. Before studying about the for loop we should have a good knowledge about the increment and decrement operators.

## 1.1 Increment and Decrement Operators

- Post Increment Operator - Expression is evaluated first using the original value of the variable and then the variable is incremented by 1. Try below code and see the output.

```
#include<stdio.h>
int main()
{
int x = 10;
int a;
a = x++;
printf("%d\n", a);
printf("%d", x);
return 0;

}
```

- Pre Increment Operator - The variable is incremented first and then the expression is evaluated using the new value of the variable. Try below code and see the output.

```
#include<stdio.h>
int main()
{
int x = 10;
int a;
a = ++x;
printf("%d\n", a);
printf("%d", x);
return 0;

}
```

- Post Decrement Operator - Expression is evaluated first using the original value of the variable and then the variable is decremented by 1. Try below code and see the output.

```
#include<stdio.h>
int main()
{
int x = 10;
int a;
a = x--;
printf("%d\n", a);
printf("%d", x);
return 0;
 }
```

- Pre Decrement Operator - The variable is decremented first and then the expression is evaluated using the new value of the variable. Try below code and see the output.

```
#include<stdio.h>
int main()
{
int x = 10;
int a;
a = --x;
printf("%d\n", a);
printf("%d", x);
return 0;
 }
```

### 1.2 for Loop Syntax

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

Try below code segment for further understanding

```
#include <stdio.h>

int main () {

   int a;


   for( a = 1; a < 10; a = a++ ){
      printf("value of a: %d\n", a);
   }

   return 0;
}
```

### 1.3 nested for loop Syntax

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

Try below code segment for further understanding

```
#include <stdio.h>

int main () {

   int a;
   int b;

   for( a = 1; a <= 5;  a++ )
   {
     for (b = 1 ; b<=5; b++)
        {

         printf("*");

        }
          printf("\n");
   }


   return 0;
}
```

## 2 while Loop

A **while** loop in C programming repeatedly executes a target statement as long as a given **condition is true**.

### 2.1 Syntax

```
while(condition) {

   statement(s);

   increment;

}
```

Try below code segment to further understanding

```
#include <stdio.h>

int main () {


   int a = 1;


   while( a < 10 ) {
      printf("value of a: %d\n", a);
      a++;
   }

   return 0;
}
```

## 3 do – while Loop

### 3.1 Syntax

A **do...while** loop is similar to a while loop, except the fact that it is guaranteed to execute at least one time.

```
do {

   statement(s);

} while( condition )
```

Try below code segment to further understanding

```c
#include <stdio.h>

int main () {


   int a = 10;

   do {
      printf("value of a: %d\n", a);
      a = a + 1;
   }
   while( a < 20 );

   return 0;
}
```

# 4 break and continue statements

## 4.1 break statement

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.

Try below code segment to further understanding.

```c
#include <stdio.h>

int main () {

   int a = 1;

   while( a < 10 ) {

      printf("value of a: %d\n", a);
      a++;

      if( a > 15)
       {

         break;
      }
   }

   return 0;
}
```

### 4.2 continue statement

The **continue** statement in C programming forces the next iteration of the loop to take place, skipping any code in between.

Try below code segment for further understanding

```c
#include <stdio.h>

int main () {

   int a = 10;

   do {

      if( a == 15) {
         a = a + 1;
         continue;
      }

      printf("value of a: %d\n", a);
      a++;

   } while( a < 20 );

   return 0;
}
```

## 5 Exercises

### 5.1 for loop

a. Develop a program to display numbers from 1 to any given number
b. Write a program to print integers from -5 to 5.
c. Write a program to print integers from 10 to 1.

d. Write C program to print the following output

i.
```
*
* *
* * *
* * * *
* * * * *
```

ii.
```
* * * * *
* * * *
* * *
* *
*
```

iii.
```
666666
55555
4444
333
22
1
```

e. Write a program by using for loop to compute the sum of

1+2+3+ . . . +n (n should be a keyboard input)

f. The factorial of an integer n is the product of consecutive integer from 1 to n. n!= n*(n-1)*…*1. Write a c program by using for loop to compute and print the factorial of any given number

## 5.2 while loop

a. Write a program to print numbers from one to given n numbers(n should be a keyboard input)
b. Write a program to print all even numbers from one to fifty.

c. Write a program by using while loop to compute the sum of 1+2+3+ . . . +n ( n should be a keyboard input)

d. You can try for loop examples by using while loop

## 5.3 do – while loop

a. Write a program to print all odd numbers from one to fifty.

b. Write a program to print numbers between 1 and 50 which are multiple of 4 using the do while loop.