Experiment No. 6
LOOPING STATEMENTS

I.OBJECTIVES

1.To employ different looping statements in C++ programs.
2. To differentiate and use looping statements in different applications.

II. LABORATORY EXERCISES

Encode the following program and compile them. Illustrate the output.
Note: if getch(); will not work on other C++ ide compiler replace it with _getch();
For www.onlinegdb.com editor and cxxdroid (mobile app) you may remove it.

Program no.1 (save as loopex1.cpp)

```
#include<iostream>
#include<windows.h>
#include<conio.h>
using namespace std;
int main ()
{
int num =1;
while (num<=5){
        cout<<"\n "<<num;
Sleep(3000);
num++;
}
_getch();
return 0;
}
```

Program No. 2 (save as loopex2.cpp)

```
#include<iostream>
#include<conio.h>
using namespace std;
int main ()
{
int x;
for (x =1; x<10;x++)
{ if (x ==5)
break;
cout<< x<< " ";
}
cout << "\nBroke out of loop at x of " << x <<endl;
_getch();
return 0;
}
```

Program No 3. (save as loopex3.cpp)

```
#include<iostream>
#include<conio.h>
using namespace std;
int main ()
{
int y, x=1, total =0;
while (x <= 10)
{ y=x * x;
cout << y << endl;
total +=y;
++x; }
cout << "Total is " <<total <<endl;
_getch();
return 0;
}
```

Program No 4. (save as loopex4.cpp)

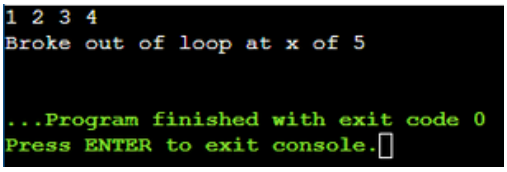
```
#include<iostream>
#include<conio.h>
using namespace std;
int main ()
{
int x, y;
cout<<"\t\t\t\t   Multiplication   Table   \n\n";
for(x=1;x<=10;x++)
{
for(y=1;y<=10;y++)
{
cout<<x*y<<"\t";
}
cout<<"\n";
}
_getch();
return 0;
}
```

III. Summary of Program Outputs

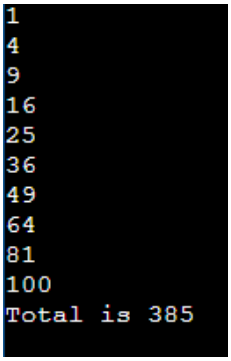
Direction: Demonstrate the corresponding output for each of the given programs. Give your observation and analysis for each of the problems.

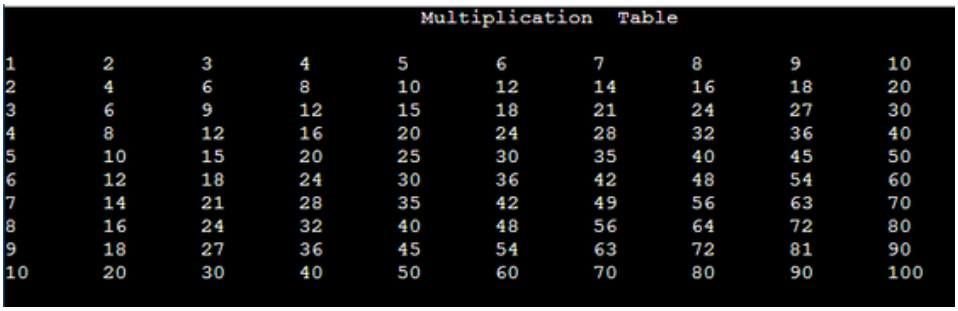
1. It displayed the sequence of numbers from 1 to 5, but it was not displayed all at once. Instead it was displayed one-by-one and with a interval of about 3 seconds. It displayed the number 1 to 5 because we initialized the value of num into 1 and we had a while condition that continues while num is less than or equals to 5.

2. In here, we can observe that we have a for loop, that starts from an initial value of 1 and has a condition that continues while x is less than 10, and it increments by 1, but we inserted a break by using a if condition that if x is equals to 5, and then we had a output of string that says where it broke out of loop.



3. In this code we can see that each number displayed is the squared value of each number from 1 up until to 10, this was achieved by using a simple multiplication operator by multiplying *x* by itself, and then the next number is achieved by incrementing *x* by 1, thus having the square of the next number. and then by having the value of the *total* variable added into *y* by each number squared, we get the total of all the number squared from 1 up until 10.





4. In this code we can see that it is a multiplication table from 1 up until 10, this was achieved by using a for loop inside a for loop, with *x* and *y* multiplied to each other and displayed in a column. And it increments by 1 so that the next multiple can be achieved.

IV. Supplementary Problems

1.Write a program which produces the given sequence numbers (in alternative arrangement).

1, 5, 2, 4, 3, 3, 4, 2, 5, 1,

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
int x, y = 5;
for(x=1;x<4;x++){
    if(x==3){
    cout<<x<<", ";
    goto loop1;
    }
    cout<<x<<", ";
    cout<<y<<", ";
    --y;
    }loop1:
    for(x=3;x>0;--x){
    cout<<x<<", ";
    ++y;
    if(x==1){
        break;
    }
    cout<<y<<", ";
    }
getch();
return 0;
}
```

```
1, 5, 2, 4, 3, 3, 4, 2, 5, 1,

...Program finished with exit code 0
Press ENTER to exit console.
```

IV. Supplementary Problems

2.Write a program that reverses the input number n. Formulate an equation to come up with the answer. The program will continue to ask a number to reverse till the user enter '0' or to quit.

Sample output:
Enter a number: 1238
Reverse number: 8321

```cpp
#include <iostream>
#include <string>
using namespace std;
int main()
{
    int x, rev_x =0, rev_2 = 0, rem;
    point1:
    cout<< "Enter a number(0 to stop): ";
    cin>> x;
    while(x!=0){
      if(x>0 && rev_2==0){
        rem = x%10;
        x = x/10;
        rev_x = (rev_x*10)+rem;
      }
      if(x==0 && rev_2==0){
        rev_2 = rev_x;
        rev_x = 0;
        goto point1;
      }
      if(x>0 && rev_2>=1){
          rem = x%10;
          x = x/10;
          rev_x = (rev_x*10)+rem;
        if(x==0){
            string n1 = to_string(rev_x);
            string n2 = to_string(rev_2);
            string n3 = n1 + n2;
            int N3 = stoi(n3);
            rev_2 = N3;
            goto point2;
        }
      }
      point2:
      if(x==0 && rev_2>=1){
        rem = 0;
        rev_x = 0;
        goto point1;
      }
    }
    cout<<"Reversed number: "<<rev_2;
return 0;
}
```

```
Enter a number(0 to stop): 1234
Enter a number(0 to stop): 4321
Enter a number(0 to stop): 0
Reversed number: 12344321

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter a number(0 to stop): 987
Enter a number(0 to stop): 654
Enter a number(0 to stop): 321
Enter a number(0 to stop): 0
Reversed number: 123456789

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter a number(0 to stop): 23
Enter a number(0 to stop): 24
Enter a number(0 to stop): 25
Enter a number(0 to stop): 0
Reversed number: 524232

...Program finished with exit code 0
Press ENTER to exit console.
```

Instructor: Engr. BIE

IV. Supplementary Problems

3.Write a program to input a number then output the Fibonacci series up to the inputted number series.

Enter a number: 10
Fibonacci series: 01 1 2 3 5 8 13 21 34

```cpp
#include <iostream>
using namespace std;
int main()
{
    int x, y, z = 0, initial_num = 0, next_num = 1;
    cout<< "Enter a number: ";
    cin>> x;
    cout<< "Fibonacci series: ";
    if(x==1){
        z = x;
        cout<<initial_num;
        }
    while(z<x){
        if(x==0){
        break;
        }
        if(x==2){
           cout<<initial_num<<" "<<next_num;
        break;
        }
        if(z==0){
            cout<<initial_num<<" ";
        z++;
        }
        if(z==1){
            cout<<next_num<<" ";
        z++;
        }
        y=initial_num + next_num;
        initial_num=next_num;
        next_num=y;
        cout<<y<<" ";
        z++;
    }
return 0;
}
```

```
Enter a number: 0
Fibonacci series:

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter a number: 1
Fibonacci series: 0

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter a number: 5
Fibonacci series: 0 1 1 2 3

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter a number: 8
Fibonacci series: 0 1 1 2 3 5 8 13

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter a number: 13
Fibonacci series: 0 1 1 2 3 5 8 13 21 34 55 89 144

...Program finished with exit code 0
Press ENTER to exit console.
```

```
Enter a number: 20
Fibonacci series: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181

...Program finished with exit code 0
Press ENTER to exit console.
```

Instructor: Engr. BIE

V. Conclusion

in this activity, I have realized that there are correct steps in building a code; firstly, is identify the set of instructions, understanding the given instructions is paramount to the success and effectivity of the code. Because in the past, what I would do is, just identify the set of instructions and build the code based on what is needed. I would skip the part where I need to formulate a logic for the structure of my code, resulting in a tedious and long hours of coding the program, i would correct the code one by one based on its output. But now, I tried building a code where I analyze the set of instructions and formulate the logic based on that. The result is a code where I can understand truly and although some of it isn't really concise. In the near future, it's probably a good idea to build a code that is concise and understandable.