Experiment No.1
Basic programming constructs like branching and looping
Date of Performance:
Date of Submission:

Aim :- To apply programming constructs of decision making and looping.

Objective :- To apply basic programming constructs like Branching and Looping for solving arithmetic problems like calculating factorial of a no entered by user at command prompt .

Theory :-

Programming constructs are basic building blocks that can be used to control computer programs. Most programs are built out of a fairly standard set of programming constructs. For example, to write a useful program, we need to be able to store values in variables, test these values against a condition, or loop through a set of instructions a certain number of times. Some of the basic program constructs include decision making and looping.

Decision Making in programming is similar to decision making in real life. In programming also we face some situations where we want a certain block of code to be executed when some condition is fulfilled. A programming language uses control statements to control the flow of execution of program based on certain conditions. These are used to cause the flow of execution to advance and branch based on changes to the state of a program.

- if
- if-else
- nested-if
- if-else-if
- switch-case
- break, continue

These statements allow you to control the flow of your program's execution based upon conditions known only during run time.

A loop is a programming structure that repeats a sequence of instructions until a specific condition is met. Programmers use loops to cycle through values, add sums of numbers, repeat functions, and many other things. ... Two of the most common types of loops are the while loop and the for loop. The different ways of looping in programming languages are

- while
- do-while

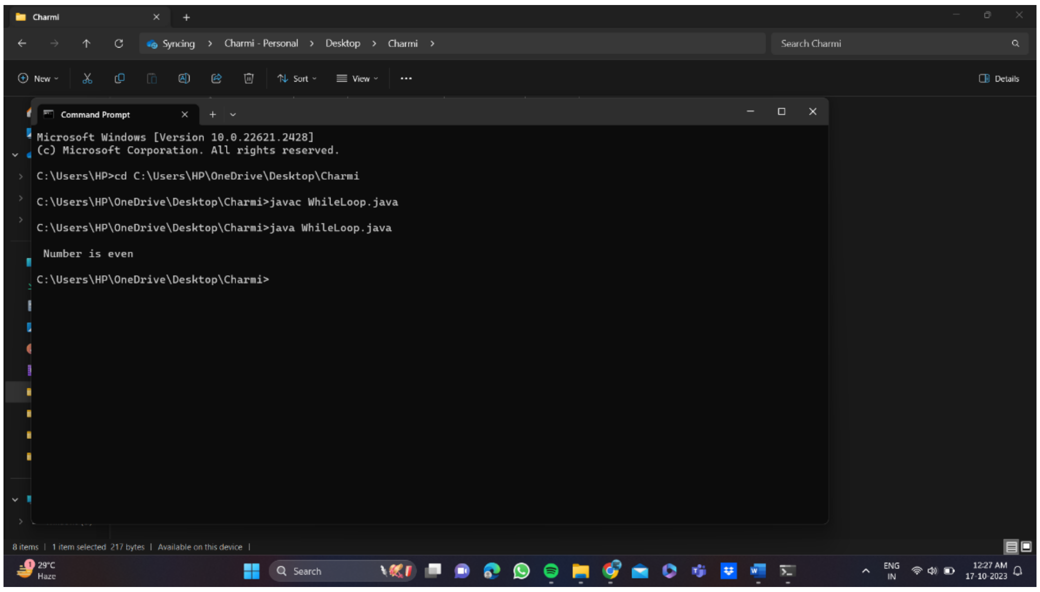
- for loop
- Some languages have modified for loops for more convenience eg :- Modified for loop in java.

For and while loop is entry-controlled loops. Do-while is an exit-controlled loop.

Code: -

1} while loop

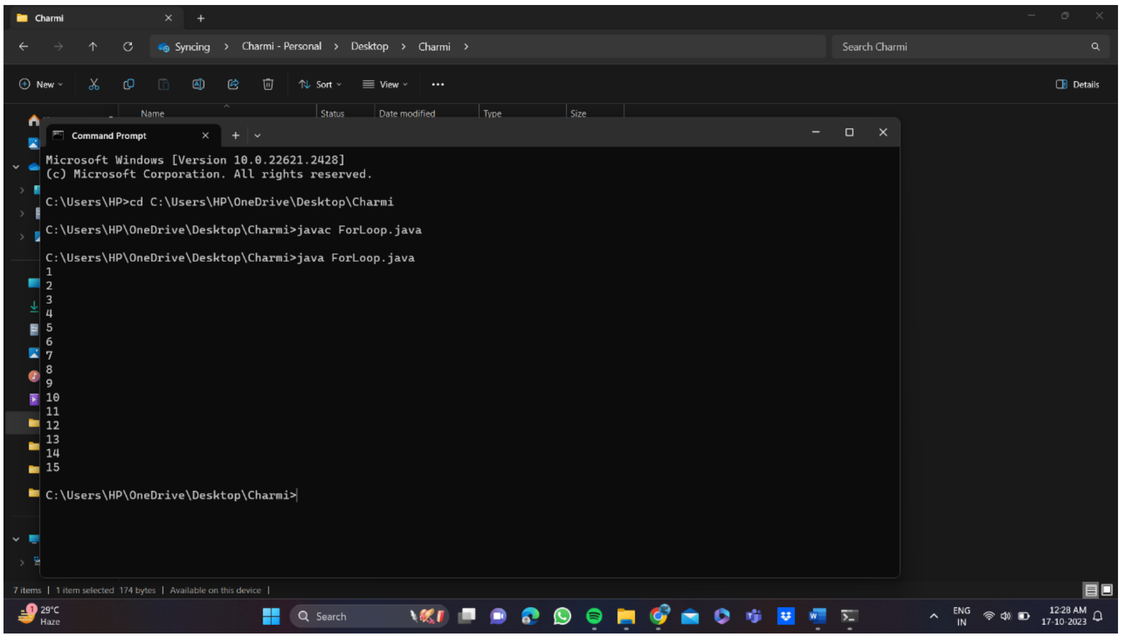
```
class Whileloop
{
  public static void main(String args[])
       {
         int a=4;
         while(a%2==0)
         {
           System.out.println("\n Number is even");
           break;
         }
       } }
```



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\Charmi

C:\Users\HP\OneDrive\Desktop\Charmi>javac WhileLoop.java

C:\Users\HP\OneDrive\Desktop\Charmi>java WhileLoop.java

 Number is even

C:\Users\HP\OneDrive\Desktop\Charmi>
```

2} for loop

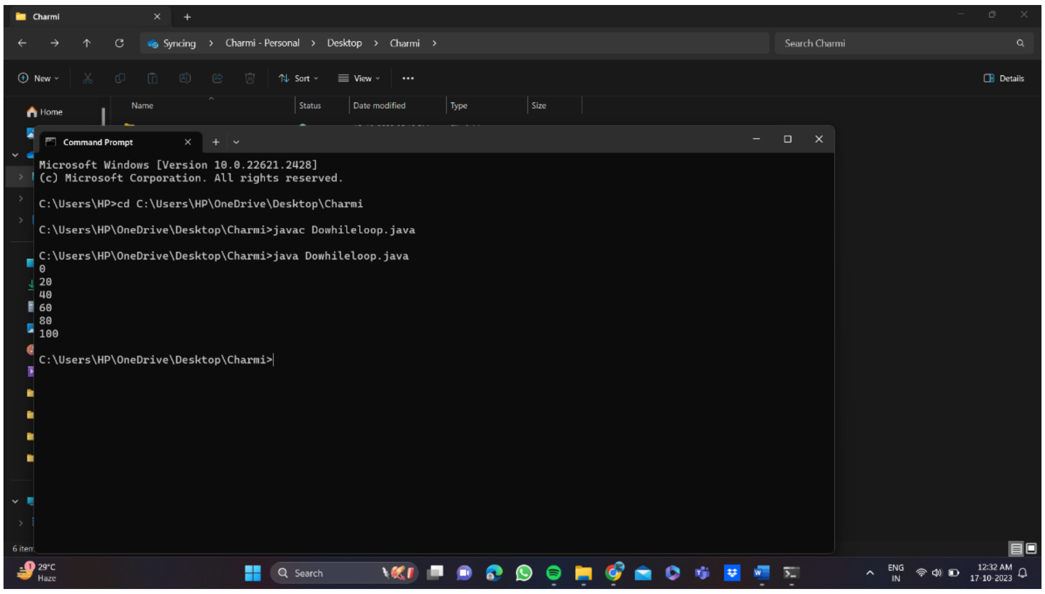
```
class Forloop
{
  public static void main(String args[])
    {
      int x;
      for(x=1;x<=10;x++)
      {
        System.out.println(x);
        }
      }
}
```



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\Charmi

C:\Users\HP\OneDrive\Desktop\Charmi>javac ForLoop.java

C:\Users\HP\OneDrive\Desktop\Charmi>java ForLoop.java
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

C:\Users\HP\OneDrive\Desktop\Charmi>
```

3} dowhile loop

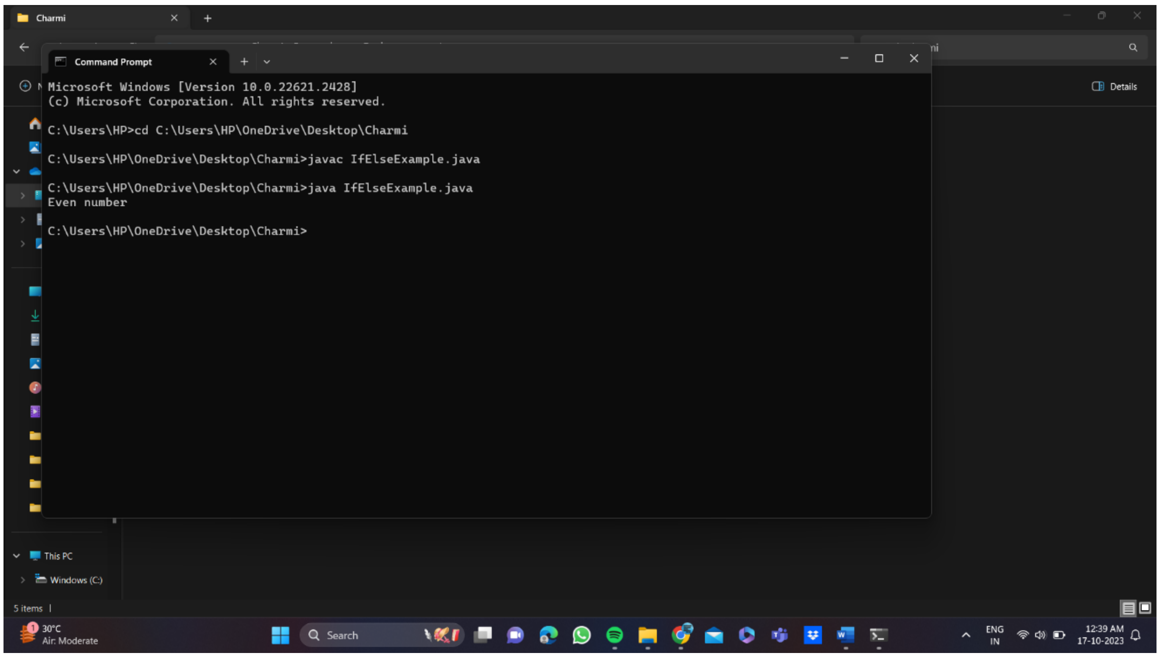
```
class Dowhileloop
{
    public static void main(String arg[])
   {
int a=0;
   do
  {
    if(a%20==0)
    {
     System.out.println(a);
     } a++;
  } while(a<=100);
  }
}
```



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\Charmi

C:\Users\HP\OneDrive\Desktop\Charmi>javac Dowhileloop.java

C:\Users\HP\OneDrive\Desktop\Charmi>java Dowhileloop.java
0
20
40
60
80
100

C:\Users\HP\OneDrive\Desktop\Charmi>
```

4}if else

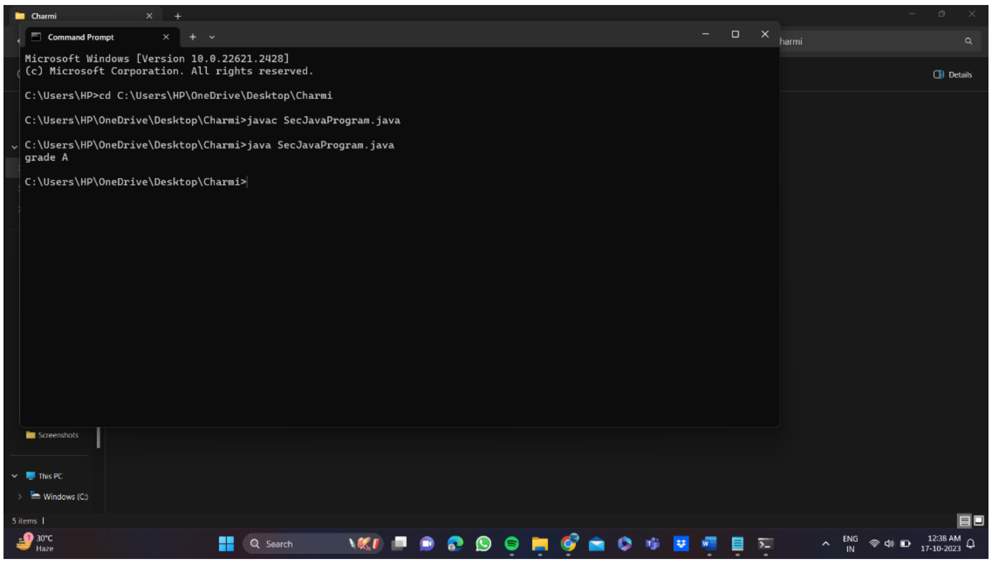
```
public class IfElseExample {
public static void main(String[] args) {
    int number=10;
        if(number%2==0){
        System.out.println("Even number");
    }else{
        System.out.println("Odd number");
    }
}
}
```



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\Charmi

C:\Users\HP\OneDrive\Desktop\Charmi>javac IfElseExample.java

C:\Users\HP\OneDrive\Desktop\Charmi>java IfElseExample.java
Even number

C:\Users\HP\OneDrive\Desktop\Charmi>
```

5} Ladder if else

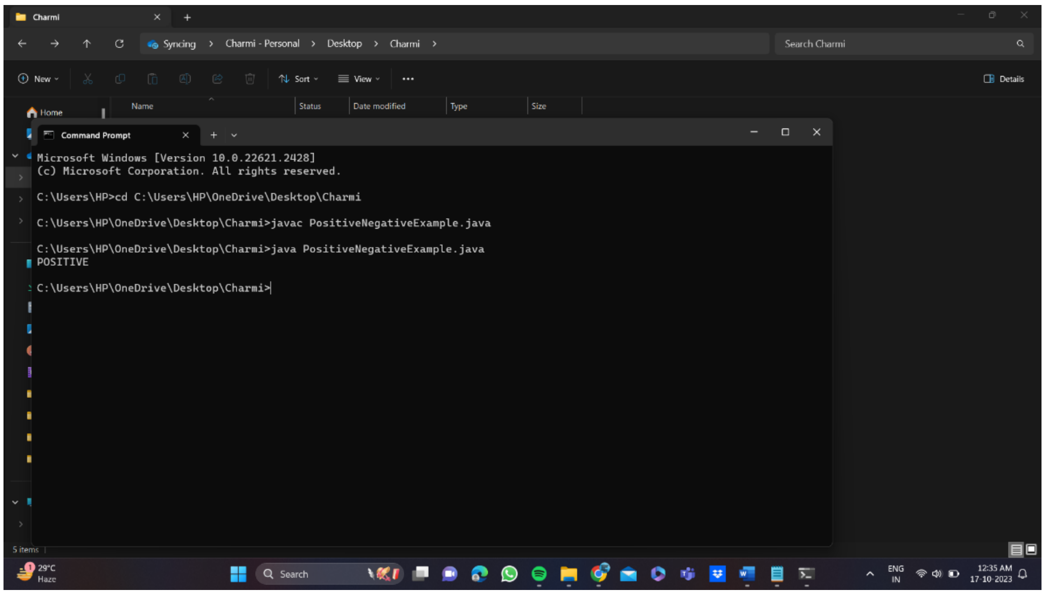
```
class SecJavaProgram
{
 public static void main(String args[])
{
 int a=90;
if(a>=90)
{
System.out.println("grade A");
}
else if(a>=80)
{
System.out.println("grade B");
}
else if(a>=70)
{
System.out.println("grade c");
}
else if(a<70)
{
System.out.println("grade F");
}
}}
```



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\Charmi

C:\Users\HP\OneDrive\Desktop\Charmi>javac SecJavaProgram.java

C:\Users\HP\OneDrive\Desktop\Charmi>java SecJavaProgram.java
grade A

C:\Users\HP\OneDrive\Desktop\Charmi>
```

6} nested if else

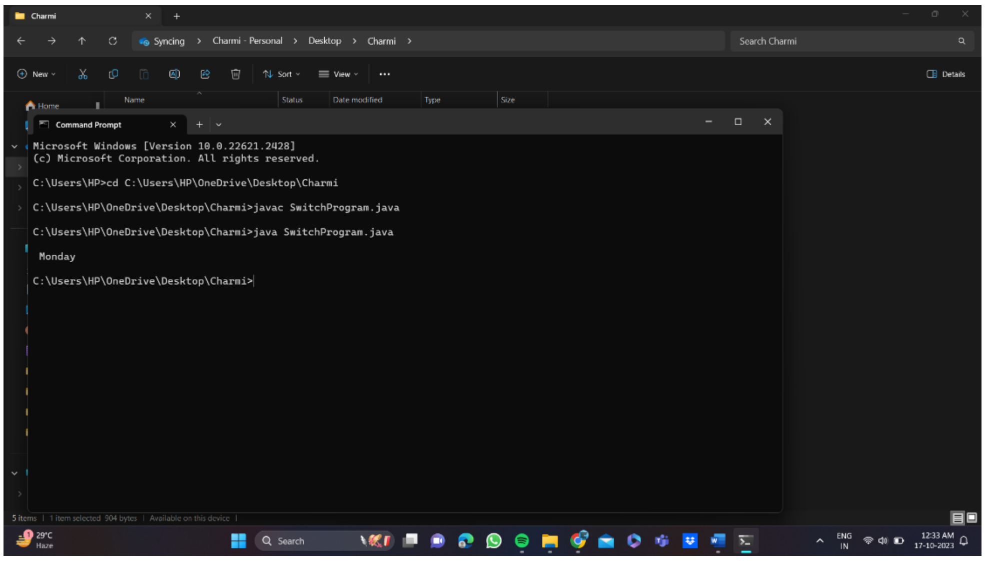
```
public class PositiveNegativeExample {
public static void main(String[] args) {
   int number=15;
   if(number>0){
   System.out.println("POSITIVE");
   }else if(number<0){
   System.out.println("NEGATIVE");
   }else{
   System.out.println("ZERO");
  }
}}
```



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\Charmi

C:\Users\HP\OneDrive\Desktop\Charmi>javac PositiveNegativeExample.java

C:\Users\HP\OneDrive\Desktop\Charmi>java PositiveNegativeExample.java
POSITIVE

C:\Users\HP\OneDrive\Desktop\Charmi>
```

7} switch

```
class SwitchProgram
{
  public static void main(String args[])
        {
         int a = 1 ;
         switch(a)
         {
         case 1 :
            System.out.println("\n Monday");
            break;
         case 2 :
            System.out.println("\n Tuesday");
            break;
         case 3 :
            System.out.println("\n Wednesday");
            break;
         case 4 :
            System.out.println("\n Thursday");
            break;
         case 5 :
            System.out.println("\n Friday");
            break;
         case 6 :
            System.out.println("\n Saturday");
            break;
         case 7 :
            System.out.println("\n Sunday");
            break;
         default :
            System.out.println("\n Not Valid");
         }
        } }
```



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\Charmi

C:\Users\HP\OneDrive\Desktop\Charmi>javac SwitchProgram.java

C:\Users\HP\OneDrive\Desktop\Charmi>java SwitchProgram.java

 Monday

C:\Users\HP\OneDrive\Desktop\Charmi>
```

Conclusion:

1) Comment on how branching and looping useful in solving problems.

Branching and looping are fundamental and powerful tools in programming that are invaluable for solving a wide range of problems. Here's how they are useful in problem-solving:

Branching (if statements):

1. Decision Making: If statements enable you to make decisions based on conditions. This is crucial for handling scenarios where different actions need to be taken depending on the state of the data or some other factor. For example, you can use if statements to handle user input, respond to specific events, or manage program flow in response to changing circumstances.

2. Error Handling: If statements are often used for error handling. You can check for exceptional conditions or error states and execute appropriate error-handling code when necessary. This ensures your program can gracefully handle unexpected situations.

Looping:

1. Repetition : Loops are essential for performing repetitive tasks efficiently. They allow you to execute a block of code multiple times, which is crucial for processing large sets of data, automating tasks, and performing iterative calculations.

2. Array and List Processing: Loops are commonly used to iterate through arrays, lists, or collections. This is fundamental for accessing and manipulating data elements within these data structures. You can, for instance, search for specific items, compute aggregates, or modify elements within an array or list using loops.