Vivekanand Education Society's
Institute of Technology
Approved by AICTE & Affiliated to University of Mumbai

Artificial Intelligence and Data Science Department.

OOPM / Odd Sem 2021-22 / Experiment.

YASH SARANG.
47 / D6AD.

EXPERIMENT - 2.

AIM: Programs on Basic programming constructs like branching and looping using functions and accepting input through the keyboard.

THEORY:

There are four different ways of reading input from the user in the command line environment.

1. Using Buffer Reader Class: This method is used by wrapping the system in an Input Stream Reader which is wrapped in a Buffered Reader, we can read input from the user in the command line. The input buffer is for efficient reading.

Syntax :

```
import java.io.BufferReader;
import java.io.soException;
import java.io.InputStreamReader;
class Test
{
public static void main(String[] args)
throws IoException
        {
                Buffered reader = new Buffered Reader (new InputStreamReader(System.in));
                String name = reader.readLine();
                System.out.println(name);
        }
}
```

2. Using Scanner Class: The main purpose of the scanner class is rare primitive types and strings using regular expressions however, it can be used. to read the input from the command line.

Convenient methods for parsing primitives(nextInt(), nextFloat(),etc.)
The reading methods are not synchronized.
Syntax :

```
import java.util.*;
Scanner in = new Scanner(System.in)
```

3. Using Console Class: It can be used for reading password–like input without echoing the characters entered by the user; the format string syntax can also be used.

Reading methods are synchronized. Format String syntax can be used.
Syntax :

```
String name = System.console().readLine();
```

4. Using Command line argument: The command line arguments are stored in the string format. The parse int method of the integer class converts string argument into an integer. Similarly float and others during execution.

Syntax :

```
for(int i = 0; i < args.length; i++)
{
        System.out.println(args[i]);
}
```

Function (static):
A static method is a method that belongs to a class rather than an instance of a class. The method is accessible to every instance of a class, but methods defined in an instance are only able to access by that object of the class. A static method is not part of the class definition. Unlike instance methods, a static method is referred to by the class name can be invoked without
creating an object of the class.

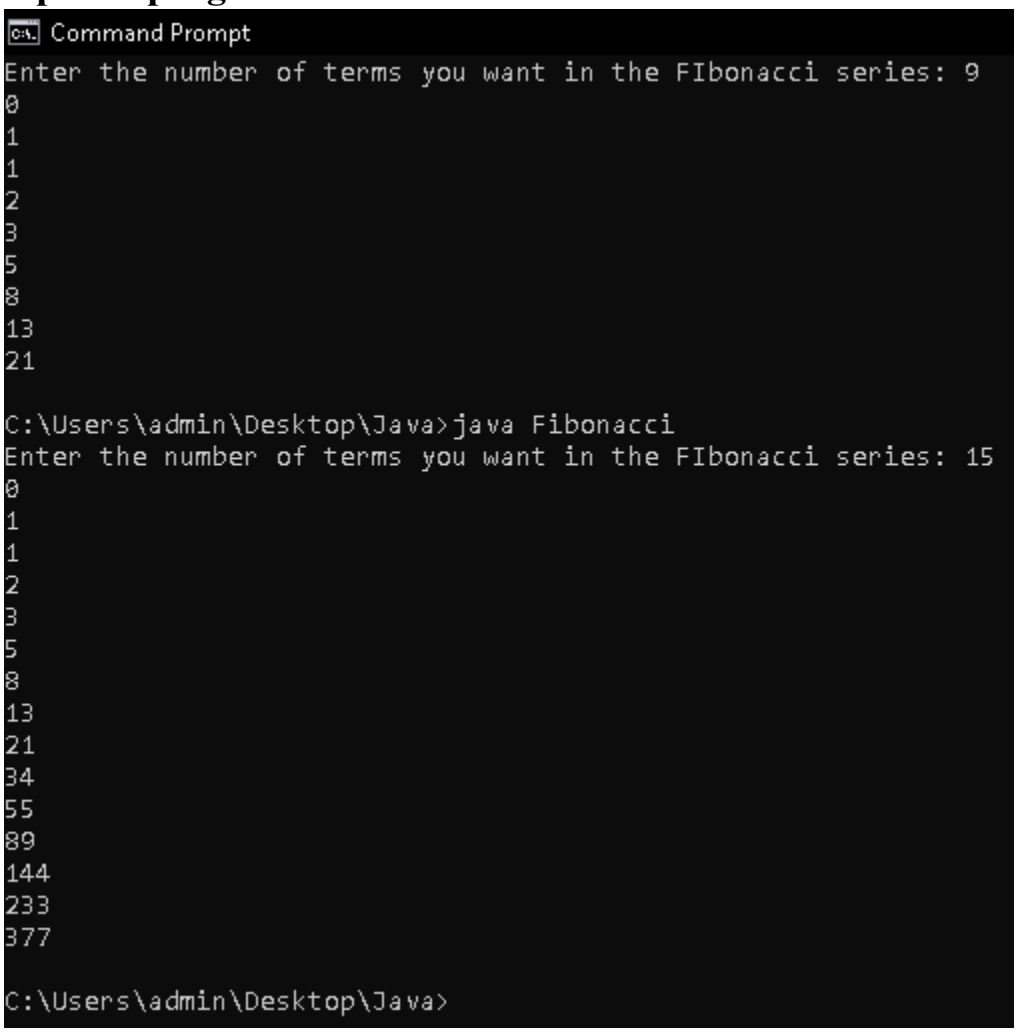
Program 1:
Print the Fibonacci series up to the nth term taking the value of n from the user.

```
import java.lang.*;
import java.util.*;
class Fibonacci
```

```
{
	public static void main(String args[])
	{
		Scanner input = new Scanner(System.in);
		System.out.print("Enter the number of terms you want in the FIbonacci series: ");
		int n = input.nextInt();

		int x[] = new int[n];
		x[0] = 0;
		x[1] = 1;
		for (int i = 2; i <= n-1; i++)
		{
		x[i] = x[i-1] + x[i-2];
		}
		for(int i=0; i<n;i++)
		{
			System.out.println(x[i]);
		}
	}
}
```

The output of program 1:



Program 2: Program to reverse any four-digit no.

```
import java.lang.*;
import java.util.Scanner;

class ReverseNumbers
{
	public static void main(String[] args)
	{
		System.out.print("\nEnter the number to be Reversed: ");
		Scanner input = new Scanner(System.in);
		int number = input.nextInt();
		int reverse = 0;
		while(number != 0)
		{
			int remainder = number % 10;
			reverse = reverse * 10 + remainder;
			number = number/10;
		}
		System.out.println("The reverse of the given number is: " + reverse);
	}
}
```

The output of program 2:

```
C:\Users\admin\Desktop\Java>java ReverseNumbers

Enter the number to be Reversed: 4536
The reverse of the given number is: 6354
```

Program 3: Program for pascal's triangle.

```
import java.lang.*;
import java.util.Scanner;

class PascalTriangle
{
	public static void main(String[] args)
	{
		Scanner input= new Scanner(System.in);
		System.out.print("\n Enter the number of rows you want in your Pascal's triangle: ");
		int n = input.nextInt();
		int arr[]= new int[n+2];
		arr[0]=1;

		System.out.println("The Pascal Triangle will be: ");
		System.out.println(arr[0]);

		for(int i = 1; i < n ; i++)
		{
```

```
                for (int j = i; j >= 1 ; j--)
                {
                        arr[j]=arr[j-1] + arr[j];
                }
                for (int k = 0; k <= i ; k++)
                {
                        System.out.print(arr[k] + " " );
                }
                System.out.print("\n");
        }

    }
}
```

The output of program 3:

```
C:\Users\admin\Desktop\Java>javac PascalTriangle.java

C:\Users\admin\Desktop\Java>java PascalTriangle

 Enter the number of rows you want in your Pascal's triangle: 6
The Pascal Triangle will be:
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1

C:\Users\admin\Desktop\Java>
```

Program 4: Program for identifying armstrong numbers.

```
import java.lang.*;
import java.util.Scanner;

class ArmstrongNumbers
{
    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in);
        System.out.print("\n Enter the number to check if it is an Armstrong Number: ");
        int num = input.nextInt();
        int temp = num;
        int soc = 0;

        while(num != 0)
        {
            int remainder = num % 10;
            System.out.println(remainder);
            soc = soc + (remainder*remainder*remainder);
```

```
                num = num/10;
            }
            if(temp == soc)
            {
                System.out.println(temp + " is an Armstrong Number.");
            }
            else
            {
                System.out.println(temp + " is not an Armstrong Number.");

            }
        }
}
```

The output of program 4:

```
C:\Users\admin\Desktop\Java>javac ArmstrongNumbers.java

C:\Users\admin\Desktop\Java>java ArmstrongNumbers

 Enter the number to check if it is an Armstrong Number: 153
3
5
1
153 is an Armstrong Number.

C:\Users\admin\Desktop\Java>java ArmstrongNumbers

 Enter the number to check if it is an Armstrong Number: 1587
7
8
5
1
1587 is not an Armstrong Number.

C:\Users\admin\Desktop\Java>java ArmstrongNumbers

 Enter the number to check if it is an Armstrong Number: 370
0
7
3
370 is an Armstrong Number.
```

Program 5: Calculate Sin and Cos values corresponding to the given value

```
import java.util.Scanner;
import java.lang.*;
import java.math.*;

class SineAndCosine
{
    public static long fact(int number)
```

```
{
long result = 1;
      for (int factor = 2; factor <= number; factor++)
         {
                  result *= factor;
         }
return result;
}
public static double Sin(double angle)
{
         double temp,sin=0.0;
         for(int i=1 ; i>=0 ; i+=4)
         {
                  temp = Math.pow(angle,i)/fact(i);
                  if(i>=25)
                           break;
                  else
                           sin = sin + temp;
         }
         for(int i=3 ; i>=0 ; i+=4)
         {
                  temp = Math.pow(angle,i)/fact(i);
                  if(i>=25)
                           break;
                  else
                           sin = sin - temp;
         }
         return sin;
}
public static double Cos(double angle)
{
         double temp,cos=0.0;
         for(int i=0 ; i>=0 ; i+=4)
         {
                  temp = Math.pow(angle,i)/fact(i);
                  if(i>=25)
                           break;
                  else
                           cos = cos + temp;
         }
         for(int i=2 ; i>=0 ; i+=4)
         {
                  temp = Math.pow(angle,i)/fact(i);
                  if(i>=25)
                           break;
                  else
                           cos = cos - temp;
         }
         return cos;
}
public static void main(String[] args)
{
```

```
        Scanner input= new Scanner(System.in);
        System.out.print("Enter the measurement of angle (in radians): ");
        double angle= input.nextDouble();

        System.out.println("Sine of " + angle + " is " + Sin(angle) + " \n Cosine of " + angle + " is
    " + Cos(angle) );
    }
}
```

The output of program 5:

```
C:\Users\admin\Desktop\Java>javac SineAndCosine.java

C:\Users\admin\Desktop\Java>java SineAndCosine
Enter the measurement of angle (in radians): 1.1
Sine of 1.1 is 0.8912073600614352
 Cosine of 1.1 is 0.45359612142557737

C:\Users\admin\Desktop\Java>java SineAndCosine
Enter the measurement of angle (in radians): 3.2
Sine of 3.2 is -0.05837420485546522
 Cosine of 3.2 is -0.9982948415388571

C:\Users\admin\Desktop\Java>java SineAndCosine
Enter the measurement of angle (in radians): 10
Sine of 10.0 is -12554.853487021188
 Cosine of 10.0 is -119626.95086524785

C:\Users\admin\Desktop\Java>
```
