Name – Prerna Sunil Jadhav **SAP ID - 60004220127**

Experiment No – 06(A)

AIM: To implement Functions, recursive functions and overloading (CO1)

THEORY:

- Functions/ Methods
  - Java Method is a collection of statements that perform some specific task and return the result to the caller.
  - A Java method can perform some specific task without returning anything.
  - Methods in Java allow us to reuse the code without retyping the code.
  - In Java, every method must be part of some class that is different from languages like C, C++, and Python.
    1. A method is like function i.e. used to expose behavior of an object.
    2. it is a set of codes that perform a particular task.
- Recursion
  - Recursion is the technique of making a function call itself. This technique provides a way to break complicated problems down into simple problems which are easier to solve.
  - Just as loops can run into the problem of infinite looping, recursive functions can run into the problem of infinite recursion. Infinite recursion is when the function never stops calling itself. Every recursive function should have a halting condition, which is the condition where the function stops calling itself.
- Method Overloading
  - If a class has multiple methods having same name but different in parameters, it is known as Method Overloading.
  - If we have to perform only one operation, having same name of the methods increases the readability of the program.

PROGRAM 1: Write A Program to check if 2 strings are Meta strings or not. Meta strings are the strings which can be made equal by exactly one swap in any of the strings. Equal string are not considered here as Meta strings.
Example: str1 = "geeks", str2 = "keegs"
By just swapping 'k' and 'g' in any of string, both will become same.
Example: str1 = "Converse", str2 = "Conserve"
By just swapping 'v' and's' in any of string, both will become same.

CODE (i): WAP to display area of square and rectangle using the concept of overloaded functions

Code1_MethodOverloading.java

```java
package Exp6;

class Shape
{
    void area(float x)
    {
        System.out.println("The area of the square is "+Math.pow(x, b: 2)+" sq units");
    }
    void area(float x, float y)
    {
        System.out.println("The area of the rectangle is "+x*y+" sq units");
    }
    void area(double x)
    {
        double z = 3.14 * x * x;
        System.out.println("The area of the circle is "+z+" sq units");
    }
}

public class Code1_MethodOverloading {
    Run | Debug
    public static void main(String[] args) {
        System.out.println(x: "Prerna Sunil Jadhav - 60004220127\n");

        Shape ob = new Shape();
        ob.area(x: 5);
        ob.area(x: 11,y: 12);
        ob.area(x: 2.5);
    }
}
```

OUTPUT:

```
Prerna Sunil Jadhav - 60004220127

The area of the square is 25.0 sq units
The area of the rectangle is 132.0 sq units
The area of the circle is 19.625 sq units
```

(ii) Write menu driven program to implement recursive functions for following tasks

CODE (ii)(a): To find GCD and LCM

Code2_GCD.java

```java
package Exp6;

public class Code2_GCD {
    Run | Debug
    public static void main(String[] args) {
        System.out.println(x: "Prerna Sunil Jadhav - 60004220127\n");
        int n1 = 4, n2 = 22;
        int gcd = gcd(n1, n2);
        int lcm = lcm(n1, n2);
        System.out.printf(format: "G.C.D of %d and %d is %d.\n", n1, n2, gcd);
        System.out.printf(format: "L.C.M of %d and %d is %d.", n1, n2, lcm);
    }
    static int lcm(int a, int b)
    {
        return (a / gcd(a, b)) * b;
    }

    public static int gcd(int n1, int n2)
    {
        if (n2 != 0)
            return gcd(n2, n1 % n2);
        else
            return n1;
    }
}
```

OUTPUT:

```
Prerna Sunil Jadhav - 60004220127

G.C.D of 4 and 22 is 2.
L.C.M of 4 and 22 is 44.
```

CODE (ii)(b): To find X^y

Code3_Power.java

```java
package Exp6;

class Code3_Power {
    Run | Debug
    public static void main(String[] args) {
        System.out.println(x: "Prerna Sunil Jadhav - 60004220127\n");
        int base = 3, powerRaised = 4;
        int result = power(base, powerRaised);
        System.out.println(base + "^" + powerRaised + "=" + result);
    }

    public static int power(int base, int powerRaised) {
        if (powerRaised != 0) {
            // recursive call to power()
            return (base * power(base, powerRaised - 1));
        } else {
            return 1;
        }
    }
}
```

OUTPUT:

```
Prerna Sunil Jadhav - 60004220127

3^4=81
```

CODE (ii)(c): To print n Fibonacci numbers

Code4_Fibo.java

Exp6 > Code4_Fibo.java > Code4_Fibo > main(String[])

```java
package Exp6;

public class Code4_Fibo {
    public static int fibonacciRecursion(int n) {
        if (n == 0) {
            return 0;
        }
        if (n == 1 || n == 2) {
            return 1;
        }
        return fibonacciRecursion(n - 2) + fibonacciRecursion(n - 1);
    }

    Run | Debug
    public static void main(String args[]) {
        System.out.println(x: "Prerna Sunil Jadhav - 60004220127\n");

        int maxNumber = 16;
        System.out.print("Fibonacci Series of " + maxNumber + " numbers: ");
        for (int i = 0; i < maxNumber; i++) {
            System.out.print(fibonacciRecursion(i) + " ");
        }
    }
}
```

OUTPUT:

```
Prerna Sunil Jadhav - 60004220127

Fibonacci Series of 16 numbers: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
```

CODE (ii)(d): To find reverse of number

Code5_Reverse.java

Exp6 > Code5_Reverse.java > Code5_Reverse > main(String[])

```java
package Exp6;

class Code5_Reverse {
    public static void Reverse(int num)
    {
        if (num < 10) {
            System.out.println(num);
            return;
        }
        else {
            System.out.print(num % 10);
            Reverse(num / 10);
        }
    }

    Run | Debug
    public static void main(String args[])
    {
        System.out.println(x: "Prerna Sunil Jadhav - 60004220127\n");
        int num = 870341009;
        System.out.print(s: "Reversed Number: ");
        Reverse(num);
    }
}
```

OUTPUT:

```
Prerna Sunil Jadhav - 60004220127

Reversed Number: 900143078
```

CODE (ii)(e): To 1+2+3+4+…….+ (n-1)+n

Code6_AddNumbers.java

```java
package Exp6;

public class Code6_AddNumbers {

    Run | Debug
    public static void main(String[] args) {
        System.out.println(x: "Prerna Sunil Jadhav - 60004220127\n");

        int number = 250;
        int sum = addNumbers(number);
        System.out.println("Sum of first " + number + " numbers = " + sum);
    }

    public static int addNumbers(int num) {
        if (num != 0)
            return num + addNumbers(num - 1);
        else
            return num;
    }
}
```

OUTPUT:

```
Prerna Sunil Jadhav - 60004220127

Sum of first 250 numbers = 31375
```

CODE (ii)(f): Calculate sum of digits of a number

Code7_SumOfDigits.java

```java
package Exp6;

public class Code7_SumOfDigits {
    private static int sumOfDigits(int num) {
        if (num == 0) {
            return 0;
        }
        return num % 10 + sumOfDigits(num / 10);
    }

    Run | Debug
    public static void main(String[] args) {
        System.out.println(x: "Prerna Sunil Jadhav - 60004220127\n");

        int result = sumOfDigits(num: 1234);
        System.out.println(result);
    }
}
```

OUTPUT:

```
Prerna Sunil Jadhav - 60004220127

10
```

CONCLUSION:

- Method overloading refers to the creation of more than one methods with the same name in the same class. Method overloading is a way that demonstrates polymorphism(or more accurately static polymorphism) in Java.
- Recursion is the process in which a function calls itself and the method that calls itself is known as a recursive function. This means that the method call statement is present in the body of the method itself.