National Institute of Technology Kurukshetra, Kurukshetra - 136119

Master Of Computer Application

MCA - 136

(2024-27)

OOP Using Java Lab

Lab coordinator: Dr. Sarika Jain

Submitted by:

Name: Jayesh Solanki

Semester: 2st

Section: G-4

Roll no: 524110055

Submitted to:

Ms. Smruti Mam

Mr. Aman Sir

Date: 3-05-2025

DECLARATION

I, **Jayesh Solanki**, hereby declare that the **OOP Using Java Lab Manual** submitted by me is my own original and independently completed work.

While I have referred to online resources and sample code snippets to enhance my understanding of the concepts, the structure, implementation, and content of the manual are based entirely on my own logic and effort.

This manual is distinct from others because:

1. I have included **screenshots of program outputs**, organized in a folder named after me, as proof of successful execution on my own system.
2. Each program is **modular, well-structured**, and developed using my personal understanding of the topics.
3. **No part** of this manual has been copied from any student or external source.
4. The manual is systematically organized with a **comprehensive table of contents** for easy navigation.

I fully understand and uphold the principles of academic integrity and take complete responsibility for the work I have submitted.

INDEX

Week 1 (To do some basic programs in JAVA)

Program 1: Write a Java program to display the message "This is my first java class".

Execute this program through the command line using javac and java.

Code:

```
public class Program1 {
   public static void main(String[] args) {
      System.out.println("This is my first java class");
   }
}
```

Output:

```
PS F:\Jayesh\JAVA> javac Program1.java
PS F:\Jayesh\JAVA> java Program1
This is my first java class
```

Program 2: Write a Java program to display the message "This is just a test". Pass these 5 strings as arguments in the main function while executing the program through cmd.

Code:

```
public class Program2 {
   public static void main(String[] args) {
      for(String word:args){
         System.out.print(word+" ");
      }
   }
}
```

Output:

```
PS F:\Jayesh\JAVA> javac Program2.java
PS F:\Jayesh\JAVA> java Program2 This is just a test
This is just a test
```

Program 3: Write a Java Program to make a frequency count of words in a given text entered by the user.

Code:

```
import java.util.Scanner;
public class Program3 {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter the text: ");
      String text = sc.nextLine();
      String[] words= text.split(" ");
      System.out.println("Total no. of words are: "+words.length);
      sc.close();
   }
}
```

Output:

```
Enter the text:
My name is jayesh solanki
Total no. of words are: 5
```

Program 4: Write a Java program that prompts the user for an integer and then prints out all prime numbers up to that integer. (use Scanner class to read input).

Code:

```
import java.util.Scanner;
public class Program4 {
   static boolean isPrime(int x){
      if(x<=1){
         return false;
      }
      for(int i=2;i<=Math.sqrt(x);i++){
         if(x%i==0){
            return false; }
      }
      return true;
   }
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter the number: ");
      int num = sc.nextInt();
      System.out.println("List of primes:");
      for(int i = 1 ; i<=num;i++){
         if(isPrime(i)){
            System.out.print(i+" ");
         }
      }
   }
}
```

Output:

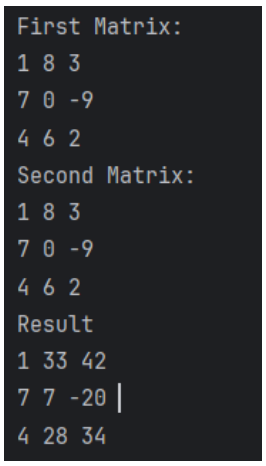
```
Enter the number: 30
List of primes:
2 3 5 7 11 13 17 19 23 29
```

Program 5: Write a Java program to multiply two given matrices.

Code:

```
public class Program5 {
   public static void main(String[] args) {
      int[][] matrix1 = {{1, 8, 3}, {7, 0, -9}, {4, 6, 2}};
      int[][] matrix2 = {{2, 7, 1}, {-7, 5, 4}, {1, 2, 3}};
      int[][] result = multiplyMatrix(matrix1, matrix2);
      System.out.println("First Matrix:");
      printMatrix(matrix1);
      System.out.println("Second Matrix:");
      printMatrix(matrix1);
      System.out.println("Result");
      printMatrix(result);
   }
   public static int[][] multiplyMatrix(int[][] mat1, int[][] mat2){
      int[][] result = new int[mat1[0].length][mat2.length];
      for(int i = 0; i<mat1.length; i++){
         for(int j = 0; j<mat2[0].length; j++){
            int res = 0;
            for(int k = 0; k<mat2.length; k++){
               res += mat1[i][k] * mat2[k][j];
               result[i][k] = res;
            }
         }
      }
      return result;
   }
   public static void printMatrix(int[][] mat){
      for(int i = 0; i< mat.length; i++){
         for(int j = 0; j< mat[i].length; j++){
            System.out.print(mat[i][j]+" ");
         }
         System.out.println();
      }
   }
}
```

Output:



Program 6: Write a Java program to find the Fibonacci series using recursive and non-recursive functions.

Code:

```
import java.util.Scanner;
public class Program6 {
   public static int recursiveFibonacci(int n) {
      if(n<=0){
         return 0;
      }
      if (n == 1)
         return n;
      return recursiveFibonacci(n - 1) + recursiveFibonacci(n - 2);
   }

   public static void nonRecursiveFibonacci(int n) {
      int a = 0, b = 1, curr;
      System.out.print(a + " " + b+" ");
      for (int i = 2; i < n; i++) {
         curr = a + b;
         System.out.print(curr+" ");
         a = b;
         b = curr;
      }
      System.out.println();
   }
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the number of terms: ");
      int n = scanner.nextInt();
      System.out.println("Using recursive method: ");
      for (int i = 0; i < n; i++) {
         System.out.print(recursiveFibonacci(i) + " ");
      }
      System.out.println();
      System.out.println("Using non recursive method: ");
      nonRecursiveFibonacci(n);
      scanner.close();
   }
```

}

Output :

```
Enter the number of terms: 15
Using recursive method:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
Using non recursive method:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

Week 2 (To study Data types, Scope, and a lifetime of variables, operators, expressions, and control statements)

Program 1: Write a Java program to identify the symbols of expression given by the user is the operator or of which datatype.

Code:

```
import java.util.Scanner;
public class Program1 {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter the expression: ");
      String str = sc.nextLine();
      for(int i=0;i<str.length();i++){
         String c="";
         String op="";
         for(int j = i; j<str.length();j++){
            if(str.substring(j, j+1).matches("[+/\\-*%]")){
               op+=str.charAt(j);
               break;
            }else if(str.charAt(j)==' ')
            {
               break;
            }
            c+=str.charAt(j);
            i++;
         }
         if(c.matches("[a-zA-Z]+")){
            System.out.println(c+" is a variable.");
         }
         else if(c.matches("[0-9]+")){
            System.out.println(c+" is an integer.");
         }
         else if(c.matches("[0-9]+\\.?[0-9]+")){
            System.out.println(c+" is a decimal.");
         }
         if(op.matches("[+\\-/*%]")){
            System.out.println(op+" is a operator.");
```

```
        }
      }
      sc.close();
   }
}
```

Output:

```
Enter the expression: abc+87*p
abc is a variable.
+ is a operator.
87 is an integer.
* is a operator.
p is a variable.
```

Program 2: Write a Java program to make a class A in that class declare function add perform addition through main

Code:

```
import java.util.Scanner;
class A{
   static double add(double a, double b){
      return a+b;
   }
}

public class Program2 {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter first number: ");
      double x = sc.nextDouble();
      System.out.print("Enter second number: ");
      double y = sc.nextDouble();
      System.out.println("Sum: "+A.add(x,y));
      sc.close();
   }
}
```

Output:

```
Enter first number: 45
Enter second number: 26
Sum: 71.0
```

Program 3: Write a Java program to print prime numbers from 1 to 50 using while & do-while loops.

Code:

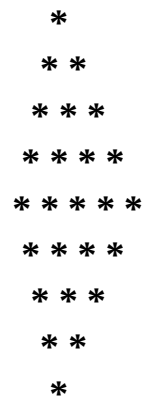
```
public class Program3 {
    static boolean isPrimeWhile(int prime) {
        if (prime <= 1) {
            return false;
        }
        int i = 2;
        while (i < prime) {
            if (prime % i == 0) {
                return false;
            }
            i++;
        }
        return true;
    }
    static boolean isPrimeDoWhile(int prime) {
        if (prime <= 1) {
            return false;
        }
        int i = 2;
        do {
            if (prime % i == 0) {
                return false;
            }
            i++;
        } while (i < prime);
        return true;
    }
    public static void main(String[] args) {
        int i = 1;
        int n = 50;
        System.out.println("//While loop");
        System.out.println("Prime numbers before " + n + " :");
        while (i <= n) {
            if (isPrimeWhile(i)) {
                System.out.println(i);
```

```
        }
        i++;
    }
    i = 1;
    System.out.println("\n//Do while loop");
    System.out.println("Prime numbers before " + n + " :");
    do {
        if (isPrimeDoWhile(i)) {
            System.out.println(i);
        }
        i++;
    } while (i <= n);
}   }
```

Output:

```
//While loop
Prime numbers before 50 :
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47
//Do while loop
Prime numbers before 50 :
3 5 7 11 13 17 19 23 29 31 37 41 43 47
```

Program 4: Write a Java program to print the following structure using for loop.



Code:

```
import java.util.Scanner;
public class Program4 {
   public static void main(String[] args) {
      System.out.print("Enter rows: ");
      Scanner sc = new Scanner(System.in);
      int n = sc.nextInt();
      int firstHalf = n%2==0?n/2:n/2+1;
      int secondHalf =n/2;
      for(int i=0;i<firstHalf;i++){
         for(int k=firstHalf-i-1;k>0;k--){
            System.out.print(" ");
         }
         for(int j=0;j<=i;j++){
            System.out.print("* ");
         }
         System.out.println();
      }
      for(int i=0;i<secondHalf;i++){
         if(secondHalf%2==0){
            System.out.print(" ");
         }
         for(int k=0;k<i;k++){
            System.out.print(" ");
         }
```

```
            for(int l=secondHalf-i;l>0;l--){
                System.out.print("* ");
            }
            System.out.println();
        }
        sc.close();
    }
}
```

Output:

Program 5: Write a Java program to find the factorial of given number using command line.

Code:

```
public class Program5 {
    static long fact(int x){
        if(x<=1){
            return 1;
        }
        return x*fact(x-1);
    }
    public static void main(String[] args) {
        int num = Integer.parseInt(args[0]);
        System.out.println("Factorial of "+num+" is "+fact(num));
```

```
    }
}
```

Output:

```
Factorial of 7 is 5040
```

Program 6: Write a Java program to find the volume of the cuboid using this pointer.

Code:

```
import java.util.Scanner;
class Cuboid {
    int length;
    int width;
    int height;
    Cuboid(int l, int w, int h) {
        this.length = l;
        this.width = w;
        this.height = h;
    }
    double volume() {
        return this.length * this.width * this.height;
    }
}

public class Program6 {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter length of Cuboid: ");
        int l = sc.nextInt();
        System.out.print("Enter width of Cuboid: ");
        int w = sc.nextInt();
        System.out.print("Enter height of Cuboid: ");
        int h = sc.nextInt();
        Cuboid c1 = new Cuboid(l, w, h);
        System.out.println("Volume of cuboid is " + c1.volume());
        sc.close();
    }
}
```

Output:

```
Enter length of Cuboid: 12
Enter width of Cuboid: 18
Enter height of Cuboid: 15
Volume of cuboid is 3240.0
```

Program 7: Write a Java program to swap the value of the speed of bikes by making class bike using call by value and call by reference.

Code:

```
import java.util.Scanner;
class Bike {
    int speed;
    Bike(int speed) {
        this.speed = speed;
    }
}
public class Program7 {
    static void swapbyvalue(int speed1, int speed2) {
        int temp = speed1;
        speed1 = speed2;
        speed2 = temp;
    }
    static void swapbyreference(Bike b1, Bike b2) {
        int temp = b1.speed;
        b1.speed = b2.speed;
        b2.speed = temp;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter speed of first bike: ");
        Bike b1 = new Bike(sc.nextInt());
        System.out.print("Enter speed of second bike: ");

        Bike b2 = new Bike(sc.nextInt());
        System.out.println("Before swapping: ");
        System.out.println("Speed of bike1: " + b1.speed);
        System.out.println("Speed of bike2: " + b2.speed);

        swapbyvalue(b1.speed, b2.speed);
        System.out.println("\nAfter swapping by value: ");
```

```
        System.out.println("Speed of bike1: " + b1.speed);
        System.out.println("Speed of bike2: " + b2.speed);

        swapbyreference(b1, b2);
        System.out.println("After swapping by reference: ");
        System.out.println("Speed of bike1: " + b1.speed);
        System.out.println("Speed of bike2: " + b2.speed);
        sc.close();
    }
}
```

Output:

```
Enter speed of first bike: 45
Enter speed of second bike: 50
Before swapping:
Speed of bike1: 45
Speed of bike2: 50

After swapping by value:
Speed of bike1: 45
Speed of bike2: 50
After swapping by reference:
Speed of bike1: 50
Speed of bike2: 45
```

Program 8: Write a Java program to find whether the number is palindrome or not using recursion.

Code:

```
import java.util.Scanner;
public class Program8 {
    static boolean isPalindrome(int num, int rev, int original) {
        if (num == 0) {
            return original == rev;
        }
        rev = rev * 10 + num % 10;
        return isPalindrome(num / 10, rev, original);
    }
    public static void main(String[] args) {
        System.out.print("Enter the number: ");
        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt();
```

```
        if (isPalindrome(num, 0, num)) {
            System.out.println(num +" is a Palindrome");
        } else {
            System.out.println(num +" is not a Palindrome");
        }
        sc.close();
    }
}
```

Output:

```
Enter the number: 4567654
4567654 is a Palindrome
```

Program 9: Write a Java program to print all the substrings of the given string "CODING".

Code:

```
import java.util.Scanner;
public class Program9 {
    static void allsubstring(String str, int n) {
        for (int i = 0; i <= str.length(); i++) {
            int j = i + n;
            if (j > str.length()) {
                break;
            }
            System.out.println(str.substring(i, i + n));
        }
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter string: ");
        String str = sc.nextLine();
        for (int i = 1; i <= str.length(); i++) {
            System.out.println("\nSubstrings of length "+i);
            allsubstring(str, i);
        }
        sc.close();
    }
}
```

Output:

```
Substrings of length 1
C
O
D
I
N
G

Substrings of length 2
CO
OD
DI
IN
NG

Substrings of length 3
COD
ODI
DIN
ING
```

```
Substrings of length 4
CODI
ODIN
DING

Substrings of length 5
CODIN
ODING

Substrings of length 6
CODING
```

Week 3 (To study the Classes and Objects in OOPJ)

Program 1: Write a Java program to create a class Rectangle with data members length and breadth. Create a method area () that finds the area of the rectangle. Use the constructor(s) to assign value to data members. Use "this" in a parameterized constructor.

Code:

```
import java.util.Scanner;
class Rectangle{
   double length;
   double breadth;
   Rectangle(double length,double breadth){
      this.length = length;
      this.breadth = breadth;
   }
   double Area(){
      return this.length*this.breadth;
   }
}
public class Program1{
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter length of rectangle: ");
      double l = sc.nextDouble();
      System.out.print("Enter breadth of rectangle: ");
```

```
        double b=sc.nextDouble();
        Rectangle rect = new Rectangle(l,b);
        System.out.println("The area of rectangle is " + rect.Area());
        sc.close();
    }
}
```

Output:

```
Enter length of rectangle: 6
Enter breadth of rectangle: 7
The area of rectangle is 42.0
```

Program 2: Print the sum, difference, and product of two complex numbers by creating a class named 'Complex' with separate methods for each operation whose real and imaginary parts are entered by the user.

Code:

```
import java.util.Scanner;
class Complex{
    double real;
    double imaginary;
    Complex(double real,double imaginary){
        this.real = real;
        this.imaginary = imaginary;
    }
    static Complex sum(Complex c1, Complex c2) {
        Complex sum = new Complex(c1.real+c2.real,c1.imaginary+c2.imaginary);
        return sum;
    }
    static Complex difference(Complex c1, Complex c2) {
        Complex diff = new Complex(c1.real-c2.real,c1.imaginary-c2.imaginary);
        return diff;
    }
    static Complex multiply(Complex c1, Complex c2){
        Complex mult=new Complex(0,0);
         mult.real = c1.real * c2.real - c1.imaginary*c2.imaginary;
         mult.imaginary = c1.real * c2.imaginary + c1.imaginary * c2.real;
        return mult;
    }
```

```
    void display(){
        System.out.println(this.real+" "+(this.imaginary>0?"+ ":"- ")+Math.abs(this.imaginary)+"i");
    }
}
public class Program2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter real part of first complex Number: ");
        double real1 = sc.nextDouble();
        System.out.print("Enter imagnary part of first complex number: ");
        double imaginary1=sc.nextDouble();
        Complex c1 = new Complex(real1,imaginary1);
        System.out.print("\nEnter real part of second complex Number: ");
        double real2 = sc.nextDouble();
        System.out.print("Enter imagnary part of second complex number: ");
        double imaginary2=sc.nextDouble();
        Complex c2 = new Complex(real2,imaginary2);
        Complex sum = Complex.sum(c1, c2);
        sum.display();
        Complex difference = Complex.difference(c1, c2);
        difference.display();
        Complex multiply = Complex.multiply(c1, c2);
        multiply.display();
        sc.close();
} }
```

Output:

```
Enter real part of first complex Number: 7
Enter imagnary part of first complex number: 9

Enter real part of second complex Number: 2
Enter imagnary part of second complex number: -5
Sum:
9.0 + 4.0i
Difference:
5.0 + 14.0i
Product:
59.0 - 17.0i
```

Program 3: Write a Java program to create a class Box with data members length, breadth, and height. Create multiple constructors to assign values to objects in different ways. Use the overridden "equals" method to compare objects, if found equal then display the objects using overridden to string method.

Code:

```
class Box {
   double length;
   double breadth;
   double height;
   Box(double l, double b, double h) {
      this.length = l;
      this.breadth = b;
      this.height = h;
   }
   Box(double l, double b) {
      this.length = l;
      this.breadth = b;
   }
   Box(double l) {
      this.length = l;
   }
   Boolean equals(Box b) {
      return this.length == b.length && this.breadth == b.breadth && this.height == b.height;
   }
   public String toString() {
      return "Length = " + this.length + ", Breadth = " + this.breadth + ", Height = " + this.height;
   }
```

```
}
public class Program3 {
      public static void main(String[] args) {
       Box b1 = new Box(5,6,7);
       Box b2 = new Box(5,6,7);
       Box b3 = new Box(20,5);
       Box b4 = new Box(16);
       if (b1.equals(b2)) {
          System.out.println("b1 and b2 are equal");
          System.out.println(b1.toString());
       } else {
          System.out.println("b1 and b2 are not equal");
       }
   }
}
```

Output:

```
b1 and b2 are equal
Length = 5.0, Breadth = 6.0, Height = 7.0
```

Week 4 (To study inheritance in Java)

Program 1: Write a Java program to identify the accessibility of a variable by means of different access specifiers within and outside the package.

Code:

```
package package1;

public class Demo {
   public int publicInt = 1;
   private int privateInt = 2;
   protected int protectedInt = 3;
   int defaultInt = 4;
}
class Access{
   public static void main(String[] args) {
      Demo A = new Demo();
      System.out.println("public "+ A.publicInt);
      // System.out.println("private "+ A.privateInt); // Error
      System.out.println("protected "+ A.protectedInt);
      System.out.println("default "+ A.defaultInt);
   }
}
```

```
public 1
protected 3
default 4
```

package package2;

```
import package1.Demo;

public class Demo2 {
   public static void main(String[] args) {
      Demo B = new Demo();
      System.out.println("public "+ B.publicInt);
      // System.out.println("private "+ B.privateInt); //Error
      // System.out.println("protected "+ B.protectedInt); //Error
      // System.out.println("default "+ B.defaultInt); //Error
   }
}
```

```
public 1
```

package package3;

```
import package1.Demo;

public class Demo3 extends Demo{
   public static void main(String[] args) {
      Demo3 B = new Demo3();
      System.out.println("public "+ B.publicInt);
      // System.out.println("private "+ B.privateInt); // Error
      System.out.println("protected "+ B.protectedInt);
      // System.out.println("default "+ B.defaultInt); // Error
   }
}
```

```
public 1
protected 3
```

Program 2: Write an employee class Marketer to accompany the other employees. Marketers make $50,000 ($10,000 more than general employees) and they have an additional method named advertise that prints "Act now, while supplies last!" Use the super keyword to interact with the Employee superclass as appropriate.

Code:

```
public class Employee {
   private double salary;
   public Employee() {
      this.salary = 40000;
   }
   public double getSalary() {
      return salary;
   }
   public void setSalary(double salary) {
      this.salary = salary;
   }
   public void work() {
      System.out.println("Employee is working");
   }
}
public class Marketer extends Employee {
   public Marketer() {
      super();
      setSalary(50000);
   }
   public void advertise() {
```

```
        System.out.println("Act now, while supplies last!");
    }
    @Override
    public void work() {
        super.work();
        System.out.println("Marketer is working");
    }
}
public class Program2 {
    public static void main(String[] args) {
        Marketer m = new Marketer();
        System.out.println("Salary: " + m.getSalary());
        m.work();
        m.advertise();
    }
}
```

Output:

```
Salary: 50000.0
Employee is working
Marketer is working
Act now, while supplies last!
```

Program 3: Create a base class Person and two derived classes as Student and Teacher with their constructors and methods. Assume the student to be in the same package as that of Person and Teacher class to be in a different package. The inheritance hierarchy would appear as follows:

a) **Add methods "get" the instance variables in the Person class. These would consist of: getName, getAge, getGender.**
b) **Add methods to "set" and "get" the instance variables in the Student class. These would consist of: getIdNum, getGPA, setIdNum.**
c) **Write a Teacher class that extends the parent class Person.**

Code:

Package p3
Person.java

```
package p3;
public class Person {
    private String name;
    private int age;
    private String gender;

    public Person(String name, int age, String gender) {
        this.name = name;
        this.age = age;
        this.gender = gender;
    }
```

```
    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public String getGender() {
        return gender;
    }
}
```

Package p3
Student.java

```
package p3;
public class Student extends Person {
    private int id;
    private double gpa;

    public Student(String name, int age, String gender) {
        super(name, age, gender);
        this.id = 0;
        this.gpa = 0.0;
    }

    public void setid(int id) {
        this.id = id;
    }

    public void setgpa(double gpa) {
        this.gpa = gpa;
    }

    public int getid() {
        return id;
    }

    public double getgpa() {
        return gpa;
    }
}
```

Package p4
Teacher.java

```
package p4;
import p3.Person;
public class Teacher extends Person {
    private String subject;
    private double salary;

    public Teacher(String name, int age, String gender, String subject, double salary) {
        super(name, age, gender);
```

```
        this.subject = subject;
        this.salary = salary;
    }

    public String getSubject() {
        return subject;
    }

    public double getSalary() {
        return salary;
    }
}
```

Program3.java

```
import p3.Student;
import p4.Teacher;
public class Program3 {
    public static void main(String[] args) {
        Student s = new Student("John Doe", 20, "male");
        s.setid(12345);
        s.setgpa(3.5);

        Teacher t = new Teacher("Jane Doe", 32, "female", "Math", 50000.0);

        System.out.println("Student Information");
        System.out.println("ID: " + s.getid());
        System.out.println("Name: " + s.getName());
        System.out.println("Age: " + s.getAge());
        System.out.println("Gender: " + s.getGender());
        System.out.println("GPA: " + s.getgpa());

        System.out.println("\nTeacher Information");
        System.out.println("Name: " + t.getName());
        System.out.println("Age: " + t.getAge());
        System.out.println("Gender: " + t.getGender());
        System.out.println("Subject: " + t.getSubject());
        System.out.println("Salary: " + t.getSalary());
    }
}
```

Output:

```
Student Information
ID: 12345
Name: Jayesh
Age: 21
Gender: M
GPA: 9

Teacher Information
Name: Jane Doe
Age: 32
Gender: F
Subject: Math
Salary: 50000
```

Program 4: Describe abstract class called Shape which has three subclasses say Triangle, Rectangle, Circle. Define one method area() in the abstract class and override this area() in these three subclasses to calculate for specific object i.e. area() of Triangle subclass should calculate area of triangle etc. Same should be for Rectangle and Circle.

Code:

```
abstract class Shape {
   abstract void area();
}

class Circle extends Shape {
   private double radius;
   public Circle(double radius) {
      this.radius = radius;
   }
```

```
    public void area() {
        System.out.println("Area of Circle: " + Math.PI * radius * radius);
    }
}

class Triangle extends Shape {
    private double base;
    private double height;
    public Triangle(double base, double height) {
        this.base = base;
        this.height = height;
    }
    public void area() {
        System.out.println("Area of Triangle: " + 0.5 * base * height);
    }
}

class Rectangle extends Shape {
    private double length;
    private double width;

    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    public void area() {
        System.out.println("Area of Rectangle: " + length * width);
    }
}

public class Program4 {
    public static void main(String[] args) {
        Circle c = new Circle(5);
        c.area();

        Triangle t = new Triangle(5, 10);
        t.area();

        Rectangle r = new Rectangle(5, 10);
        r.area();
    }
}
```

Output:

```
Area of Circle: 78.53981633974483
Area of Triangle: 25.0
Area of Rectangle: 50.0
```

Program 5: Assume that a bank maintains two kinds of accounts for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

Create a class Account that stores customer name, account number and type of account. From this derive the classes Curr-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

- **Accept deposit from a customer and update the balance.**
- **Display the balance.**
- **Compute and deposit interest.**
- **Permit withdrawal and update the balance.**
- **Check for the minimum balance, impose penalty, if necessary and update the balance.**

Code:

```
class Account {
    private int accountNumber;
    private String accountHolderName;
    protected double balance;
    protected String type;

    public Account(int accountNumber, String accountHolderName, double balance) {
        this.accountNumber = accountNumber;
        this.accountHolderName = accountHolderName;
        this.balance = balance;
    }

    public double interest() {
        return 0;
    }

    public void deposit(double amount) {
        balance += amount;
    }

    public void withdraw(double amount) {
        if (balance < amount) {
            System.out.println("Insufficient balance");
        } else {
            balance -= amount;
        }
    }

    public void display() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Account Holder Name: " + accountHolderName);
        System.out.println("Balance: " + balance);
    }
}

class SavingsAccount extends Account {
    public SavingsAccount(int accountNumber, String accountHolderName, double balance) {
        super(accountNumber, accountHolderName, balance);
        super.type = "Savings";
    }

    public double interest() {
        return this.balance * (5.0 / 100.0);
    }
}

public class Program5 {
    public static void main(String[] args) {
        SavingsAccount sa = new SavingsAccount(12345, "John Doe", 1000);
        sa.deposit(500);
        sa.withdraw(200);
        sa.display();
        System.out.println("Interest: " + sa.interest());
```

```
    }
}
```

Output:

```
Account Number: 12345
Account Holder Name: John Doe
Balance: 1300.0
Interest: 65.0
```

Week 5 (To study Interfaces and Exception Handling in Java)

Program 1: Write a Java program to compute the area of circle and rectangle using Interfaces.

Code:

```
interface Shape {
    double calculateArea();
}
class Circle implements Shape {
    private double radius;
    public Circle(double radius) {
        this.radius = radius;
    }
    public double calculateArea() {
```

```
        return Math.PI * radius * radius;
    }
}
class Rectangle implements Shape {
    private double length;
    private double width;
    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }
    public double calculateArea() {
        return length * width;
    }
}
public class Program1 {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        Rectangle rectangle = new Rectangle(4, 6);
        System.out.println("Area of Circle: " + circle.calculateArea());
        System.out.println("Area of Rectangle: " + rectangle.calculateArea());
    }
}
```

Output:

```
Area of Circle: 78.53981633974483
Area of Rectangle: 24.0
```

Program 2: Write a Java program that uses interface for the implementation of fixed-size and dynamic-size stacks (for dynamic stack size should be redefined as per the number of elements).

Code:

```
interface Stack {
    void push(int item);
    int pop();
    boolean isEmpty();
    boolean isFull();
}
class FixedStack implements Stack {
    private int[] stack;
    private int top;
    private int size;
    public FixedStack(int size) {
        this.size = size;
        stack = new int[size];
```

```
        top = -1;
    }
    public void push(int item) {
        if (!isFull()) {
            stack[++top] = item;
        } else {
            System.out.println("Stack Overflow");
        }
    }
    public int pop() {
        if (!isEmpty()) {
            return stack[top--];
        }
        System.out.println("Stack Underflow");
        return -1;
    }
    public boolean isEmpty() {
        return top == -1;
    }
    public boolean isFull() {
        return top == size - 1;
    }
}
class DynamicStack implements Stack {
    private int[] stack;
    private int top;
    public DynamicStack() {
        stack = new int[1];
        top = -1;
    }
    public void push(int item) {
        if (isFull()) {
            int[] temp = new int[stack.length * 2];
            System.arraycopy(stack, 0, temp, 0, stack.length);
            stack = temp;
        }
        stack[++top] = item;
    }

    public int pop() {
        if (!isEmpty()) {
            return stack[top--];
        }
        System.out.println("Stack Underflow");
        return -1;
    }
    public boolean isEmpty() {
        return top == -1;
    }
    public boolean isFull() {
        return top == stack.length - 1;
    }
}
```

```
public class Program2 {
   public static void main(String[] args) {
      Stack fs = new FixedStack(4);
      fs.push(1);
      fs.push(2);
      fs.push(3);
      fs.push(4);
      fs.push(5);
      System.out.println("Popped from fixed stack: " + fs.pop());
      Stack ds = new DynamicStack();
      ds.push(1);
      ds.push(2);
      ds.push(3);
      ds.push(4);
      System.out.println("Popped from dynamic stack: " + ds.pop());
   }
}
```

Output:

```
Stack Overflow
Popped from fixed stack: 4
Popped from dynamic stack: 4
```

Program 3: Write a Java program for exception handling with

StringIndexOutOfBound exception.

● Create an object of the class having StringIndexOutOfBound exception whenever an index is invoked of a string, which is not in the range.

● Each character of a string object is stored in a particular index starting from 0.

● To get a character present in a particular index of a string you can use a method charAt(int) of java.lang.String where int argument is the index.

Code:

```
public class Program3 {
   public static void main(String[] args) {
      try {
         String str = "Hello World";
         System.out.println("String: " + str);
         System.out.println("Character at index 0: " + str.charAt(0));
```

```
            System.out.println("Character at index 5: " + str.charAt(5));
            System.out.println("Character at index 20: " + str.charAt(20));
        } catch (StringIndexOutOfBoundsException e) {
            System.out.println("\nException occurred!");
            System.out.println("Error Message: " + e.getMessage());
            System.out.println("String index is out of range.");
        }
    }
}
```

Output:

```
String: Hello World
Character at index 0: H
Character at index 5:

Exception occurred!
Error Message: Index 20 out of bounds for length 11
String index is out of range.
```

Program 4: An array is declared with 5 elements. Then the code tries to access the 6th element of the array which throws an exception. Write the program for this.

Code:

```
public class Program4 {
    public static void main(String[] args) {
        int[] numbers = new int[5];
        for (int i = 0; i < 5; i++) {
            numbers[i] = i + 1;
        }
        try {
            System.out.println("Attempting to access element at index 5: " + numbers[5]);
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Exception caught: " + e.getMessage());
            System.out.println("Array index out of bounds! Array length is 5, cannot access index 5.");
        }
    }
}
```

Output:

```
Exception caught: Index 5 out of bounds for length 5
Array index out of bounds! Array length is 5, cannot access index 5.
```

Program 5: Write a suitable program for the following conditions:

a) A try block followed by multiple catch blocks

b) Catching multiple type of exceptions

c) Using throws/throw keywords

d) Using finally block

e) Using try-with-resources

f) User-defined exceptions

Code:

```
import java.io.*;
class InvalidAgeException extends Exception {
    public InvalidAgeException(String message) {
        super(message);
```

```
    }
}
public class Program5 {
    public static void checkAge(int age) throws InvalidAgeException {
        if (age < 0) {
            throw new InvalidAgeException("Age cannot be negative");
        }
    }
    public static void main(String[] args) {
        try {
            int[] numbers = { 1, 2, 3 };
            System.out.println(numbers[5]);
            int result = 10 / 0;
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Array index error: " + e.getMessage());
        } catch (ArithmeticException e) {
            System.out.println("Arithmetic error: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("General error: " + e.getMessage());
        }finally{
            System.out.println("Finally executed.");
        }
        try {
            checkAge(-5);
        } catch (InvalidAgeException e) {
            System.out.println("Invalid age error: " + e.getMessage());
        }
    }
}
```

Output:

```
Array index error: Index 5 out of bounds for length 3
Finally executed.
Invalid age error: Age cannot be negative
```

Week 6 (To study about Multi-Threading in Java)

Program 1: Suppose there are 5 workers who carry out work at the same time. Develop a program that shows this scenario with the concept of multithreading.

Code:

```
class Worker extends Thread {
    private int id;
    private int tasks;
    Worker(int id,int tasks){
        this.id=id;
        this.tasks=tasks;
    }
    public void run(){
```

```
        try {
            System.out.println("Worker "+this.id+" is working.");
            for(int i=0;i<tasks;i++){
                System.out.println("Worker "+this.id+" is completed task"+(i+1));
            }
            System.out.println("Worker "+this.id+" has completed its work.");
        } catch (Exception e) {
            // TODO: handle exception
        }
    }
}
public class Program1 {
    public static void main(String[] args) {
        int n=5;
        for(int i=0;i<n;i++){
            Worker obj = new Worker(i+1,3);
            obj.start();
        }
    }
}
```

Output:

```
Worker 3 is working.
Worker 2 is working.
Worker 1 is working.
Worker 3 is completed task1
Worker 3 is completed task2
Worker 3 is completed task3
Worker 3 has completed its work.
Worker 1 is completed task1
Worker 1 is completed task2
Worker 1 is completed task3
Worker 1 has completed its work.
```

Program 2: Write a Java program that creates three threads. The first thread displays "Good Morning" every second, the second thread displays "Hello" every two seconds, and the third thread displays "Welcome" every three seconds.

Code:

```
class MyThread extends Thread{
    private int time;
    private String message;
    MyThread(String message, int time){
        this.message=message;
        this.time=time;
    }
    public void run(){
        try {
            System.out.println("Thread has started.");
```

```
            while (true) {
                System.out.println(this.message);
                Thread.sleep(time*1000);
            }
        } catch (Exception e) {
            // TODO: handle exception
        }
    }
}
public class Program2 {
    public static void main(String[] args) {
        MyThread t1=new MyThread("GoodMorning", 1);
        MyThread t2=new MyThread("Hello", 2);
        MyThread t3=new MyThread("Welcome", 3);
        t1.start();
        t2.start();
        t3.start();
    }
}
```

Output:

```
Hello
GoodMorning
Welcome
GoodMorning
Hello
GoodMorning
GoodMorning
Welcome
Hello
```

Program 3: Implement a class that checks whether a given number is prime using both the Thread class and Runnable interface.

Code:

```
import java.util.Scanner;
class CheckPrime extends Thread{
    private int number;
    CheckPrime(int n){
        this.number=n;
    }
    public void run(){
         for(int i=2;i<Math.sqrt(number);i++){
            if(this.number%i==0){
                System.out.println(number+" is not prime.");
```

```
                return;
            }
        }
         System.out.println(number+" is prime.");
    }
}
class CheckPrime2 implements Runnable{
    private int number;
    CheckPrime2(int n){
        this.number=n;
    }
    public void run(){
        for(int i=2;i<Math.sqrt(number);i++){
            if(this.number%i==0){
                System.out.println(number+" is not prime.");
                return;
            }
        }
         System.out.println(number+" is prime.");
    }
}
public class Program3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number: ");
        int n=sc.nextInt();
        CheckPrime t1 = new CheckPrime(n);
        Thread t2 = new Thread(new CheckPrime2(n));
        t1.start();
        t2.start();
    }
}
```

Output:

```
Enter the number: 7
7 is prime.
7 is prime.
```

Program 4: Write a Java program that correctly implements the producer-consumer problem using the concept of inter-thread communication.

Code:

```
class Buffer{
    private int[] buffer;
    private int count;
    Buffer(int size){
        buffer=new int[size];
        count=0;
    }
    public synchronized void produce(int item) throws InterruptedException{
        while(count==buffer.length){
```

```
            System.out.println("Buffer is full");
            wait();
        }
        buffer[count++]=item;
        notify();
        System.out.println("Produced: "+ item);
    }
    public synchronized int consume() throws InterruptedException{
        while(count==0){
            System.out.println("Buffer is empty.");
            wait();
        }
        int item=buffer[0];
        for(int i=0;i<count-1;i++){
            buffer[i]=buffer[i+1];
        }
        count--;
        notify();
        return item;
    }
}
class Producer extends Thread{
    private Buffer buffer;
    Producer(Buffer buffer){
        this.buffer=buffer;
    }
    public void run(){
        int item=1;
        while (true) {
            try {
                buffer.produce(item);
                item++;
                Thread.sleep(1500);
            } catch (Exception e) {
                // TODO: handle exception
            }
        }
    }
}
class Consumer extends Thread{
    private Buffer buffer;
    Consumer(Buffer buffer){
        this.buffer=buffer;
    }
    public void run(){
        while (true) {
            try {
                System.out.println("Consumed: "+buffer.consume());
                Thread.sleep(1000);
            } catch (Exception e) {
                // TODO: handle exception
            }
        }
```

```
    }
}
public class Program4 {
    public static void main(String[] args) {
        Buffer buffer = new Buffer(5);
        Producer producer1 = new Producer(buffer);
        Producer producer2 = new Producer(buffer);
        Consumer consumer1 = new Consumer(buffer);
        // Consumer consumer2 = new Consumer(buffer);
        producer1.start();
        producer2.start();
        consumer1.start();
    }
}
```

Output:

```
Produced: 1
Consumed: 1
Produced: 1
Consumed: 1
Produced: 2
Produced: 2
Consumed: 2
Produced: 3
Consumed: 2
Produced: 3
Consumed: 3
Produced: 4
```

Week 7 (To study Strings in Java)

Program 1: Write a Java program to find whether a given string is palindrome or not.

Code:

```
import java.util.Scanner;
public class Program1 {
    public static boolean isPalindrome(String s){
        int i=0;
        int j=s.length()-1;
        while(i<j){
```

```
            if(s.charAt(i)!=s.charAt(j)){
                return false;
            }
            i++;
            j--;
        }
        return true;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the string: ");
        String str = sc.nextLine();
        str=str.toLowerCase().replaceAll("[ _.,!?]", "");
        System.out.println("The given string is "+ (isPalindrome(str)?"":"not ")+"a palindrome");
        sc.close();
    }
}
```

Output:

```
Enter the string:
jayeyaj
The given string is a palindrome
```

Program 2: Write a method that will remove given character from the String.

Code:

```
import java.util.Scanner;
public class Program2 {
    public static String removeChars(String s,char ch){
        return s.replaceAll(""+ch, "");
    }
    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        System.out.println("Enter the string: ");
        String str = sc.nextLine();
```

```
        System.out.print("Enter the character to remove: ");
        char ch = sc.next().charAt(0);
        System.out.println("String after removing "+ch+" is "+removeChars(str, ch));
        sc.close();
    }
}
```

Output:

```
Enter the string:
jayesh solanki
Enter the character to remove: a
String after removing a is jyesh solnki
```

Program 3: Write a Java program for sorting a given list of names.

Code:

```
import java.util.Scanner;
public class Program3 {
    public static void sortNames(String[] s){
        int n=s.length;
        for(int i=0;i<n-1;i++){
            boolean swap = false;
          for(int j=0;j<n-i-1;j++){
            if(s[j].compareTo(s[j+1])>0){
                String temp = s[j];
```

```
            s[j]=s[j+1];
            s[j+1]=temp;
            swap=true;
          }
       }
       if(!swap) return;
       }
    }
    public static void main(String[] args) {
       Scanner sc= new Scanner(System.in);
       System.out.println("Enter no of names: ");
       int n = sc.nextInt();
       String[] strArr = new String[n];
       sc.nextLine();
       for(int i=0;i<n;i++){
          System.out.println("Enter name "+(i+1)+" : ");
          strArr[i] = sc.nextLine();
       }
       sortNames(strArr);
       System.out.println("Names after sorting: ");
       for(int i=0;i<n;i++){
          System.out.println(strArr[i]);
       }
       sc.close();
    }
}
```

Output:

```
Enter no of names:
4
Enter name 1 :
Shru
Enter name 2 :
Jayesh
Enter name 3 :
Priyanshu
Enter name 4 :
Sanjay
Names after sorting:
Jayesh
Priyanshu
Sanjay
Shru
```

Program 4: Write a Java program that computes your initials from your full name and displays them.

Code:

```
import java.util.Scanner;

public class Program4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the name: ");
        String str = sc.nextLine();
        String[] strArr = str.split(" ");
        for(String name: strArr){
            System.out.print(name.charAt(0)+". ");
        }
        sc.close();
    }
}
```

Output:

```
Enter the name:
Jayesh Solanki
J. S.
```

Program 5: An anagram is a word or a phrase made by transposing the letters of another word or phrase; for example, "parliament" is an anagram of "partial men," and "software"

is an anagram of "swear oft." Write a program that figures out whether one string is an anagram of another string. The program should ignore white space and punctuation.

Code:

```
import java.util.Scanner;
public class Program5 {
   public static boolean isAnagram(String str1,String str2){
      str1=str1.toLowerCase().replaceAll("[ _.,!?]", "");
      String s = str2.toLowerCase().replaceAll("[ _.,!?]", "");
      int[] arr = new int[256];
      if(str1.length()!=s.length()) return false;
      for(int i=0;i<str1.length();i++){
         arr[str1.charAt(i)]++;
         arr[str2.charAt(i)]--;
      }
      for(int i=0;i<str1.length();i++){
         if(arr[i]!=0) return false;
      }
      return true;
   }
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter First String: ");
      String str1 = sc.nextLine();
      System.out.println("Enter Second String: ");
      String str2 = sc.nextLine();
      if(isAnagram(str1, str2)){
         System.out.println("Strings are anagram");
      }else{
         System.out.println("Strings are not anagram");
      }
      sc.close();
   }
}
```

Output:

```
Enter First String:
jayesh
Enter Second String:
yeajhs
Strings are anagram
```

Week 8 (To study Event Handling & AWT in Java)

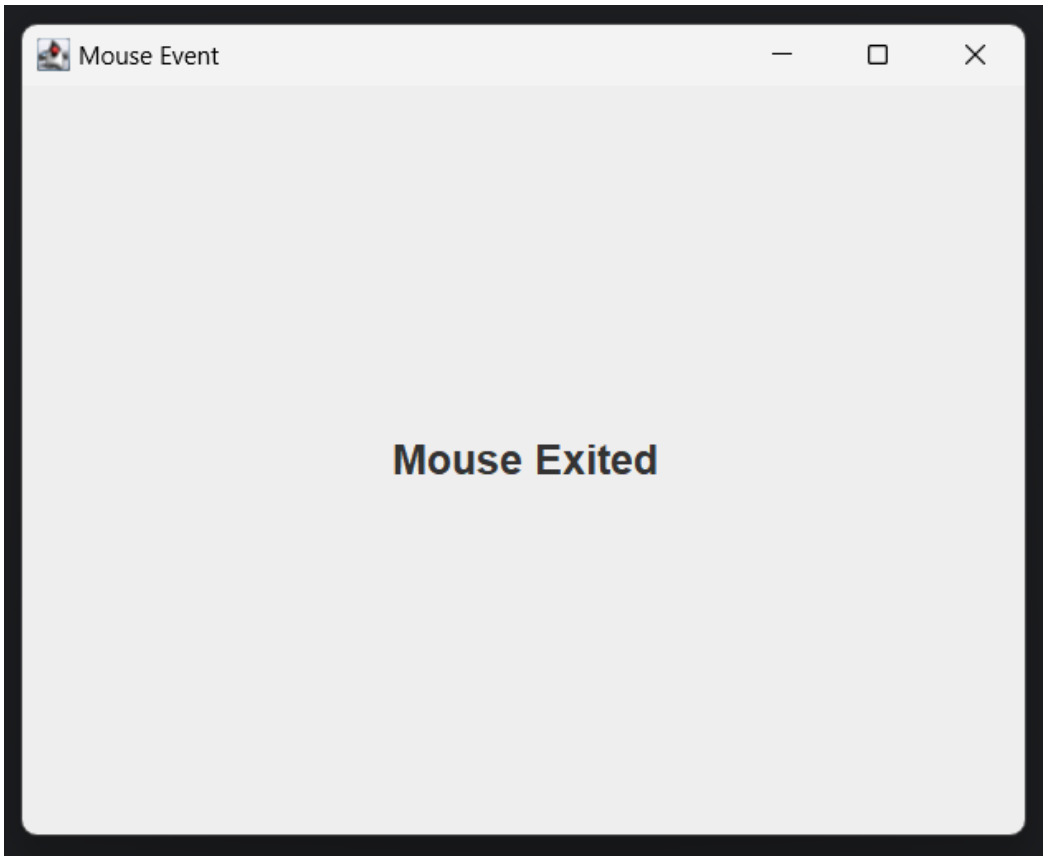
Program 1: Write a Java program that handles all mouse events and shows the event name at the centre of the window when a mouse event is fired (Use Adapter classes).

Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Program1 extends JFrame {
   private JLabel messageLabel;

   public Program1() {
      // Setup the window
      setTitle("Mouse Event");
      setSize(500, 400);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setLayout(new BorderLayout());

      // Create a label to display the event name
      messageLabel = new JLabel("No mouse event yet", JLabel.CENTER);
      messageLabel.setFont(new Font("Arial", Font.BOLD, 20));
      add(messageLabel, BorderLayout.CENTER);

      // Add mouse listeners using adapter classes
      addMouseListener(new MouseAdapter() {
         @Override
         public void mouseClicked(MouseEvent e) {
            messageLabel.setText("Mouse Clicked");
         }

         @Override
         public void mousePressed(MouseEvent e) {
            messageLabel.setText("Mouse Pressed");
         }

         @Override
         public void mouseReleased(MouseEvent e) {
            messageLabel.setText("Mouse Released");
         }

         @Override
         public void mouseEntered(MouseEvent e) {
            messageLabel.setText("Mouse Entered");
         }

         @Override
         public void mouseExited(MouseEvent e) {
            messageLabel.setText("Mouse Exited");
         }
      });

      // Add mouse motion listener using adapter class
```

```
        addMouseMotionListener(new MouseMotionAdapter() {
            @Override
            public void mouseMoved(MouseEvent e) {
                messageLabel.setText("Mouse Moved");
            }

            @Override
            public void mouseDragged(MouseEvent e) {
                messageLabel.setText("Mouse Dragged");
            }
        });
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            Program1 app = new Program1();
            app.setVisible(true);
        });
    }
}
```

Output:



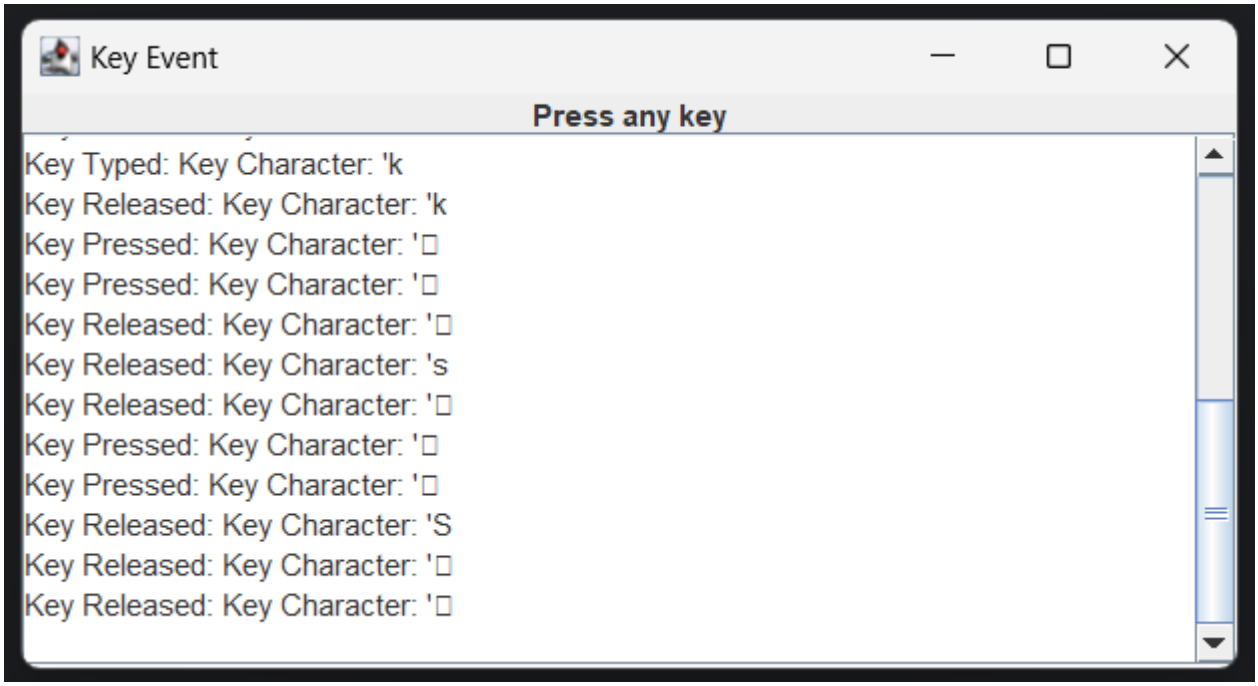
Program 2: Write a Java program for handling Key events.

Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Program2 extends JFrame implements KeyListener {
    private JTextArea displayArea;

    public Program2() {
        setTitle("Key Event");
        setSize(500, 500);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        displayArea = new JTextArea();
        displayArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(displayArea);
        add(scrollPane, BorderLayout.CENTER);

        JLabel instructionLabel = new JLabel("Press any key", JLabel.CENTER);
        add(instructionLabel, BorderLayout.NORTH);

        addKeyListener(this);

        setFocusable(true);

        setLocationRelativeTo(null);
        setVisible(true);
    }

    @Override
    public void keyTyped(KeyEvent e) {
        displayInfo("Key Typed", e);
    }

    @Override
    public void keyPressed(KeyEvent e) {
        displayInfo("Key Pressed", e);
    }

    @Override
    public void keyReleased(KeyEvent e) {
        displayInfo("Key Released", e);
    }

    private void displayInfo(String eventType, KeyEvent e) {
        String keyInfo = eventType + ": " +
                    "Key Character: '" + e.getKeyChar() + "'\n";

        displayArea.append(keyInfo);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new Program2());
    }
```

}

Output:



Program 3: Write a java program that simulates a traffic light. The program lets the user select one of three lights: Red, Yellow or Green with radio buttons. On selecting a button an appropriate message with "STOP" or "READY" or "GO" should appear above the buttons in selected color. Initially, there is no message shown.

Code:

```
import javax.swing.*;
import java.awt.*;

public class Program3 extends JFrame {
    private JLabel messageLabel;
    private JRadioButton redButton, yellowButton, greenButton;

    public Program3() {
        super("Traffic Light Simulator");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(500, 500);
        setLayout(new BorderLayout());

        messageLabel = new JLabel("", JLabel.CENTER);
        messageLabel.setFont(new Font("Arial", Font.BOLD, 20));
        add(messageLabel, BorderLayout.CENTER);

        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new FlowLayout());

        redButton = new JRadioButton("Red");
        yellowButton = new JRadioButton("Yellow");
        greenButton = new JRadioButton("Green");

        ButtonGroup buttonGroup = new ButtonGroup();
        buttonGroup.add(redButton);
        buttonGroup.add(yellowButton);
        buttonGroup.add(greenButton);

        redButton.addActionListener(e -> {
            messageLabel.setText("STOP");
            messageLabel.setForeground(Color.RED);
        });

        yellowButton.addActionListener(e -> {
            messageLabel.setText("READY");
            messageLabel.setForeground(Color.YELLOW);
        });

        greenButton.addActionListener(e -> {
            messageLabel.setText("GO");
            messageLabel.setForeground(Color.GREEN);
        });

        buttonPanel.add(redButton);
        buttonPanel.add(yellowButton);
        buttonPanel.add(greenButton);

        add(buttonPanel, BorderLayout.SOUTH);

        setLocationRelativeTo(null);
```

```
        setVisible(true);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new Program3());
    }
}
```

Output:

Week 10 (To study Swing and JDBC in Java)

Program1: Write a java programs to find factorial of a number. user is allowed to enter a number into the text field whose factorial is to be determined. On pressing the button the

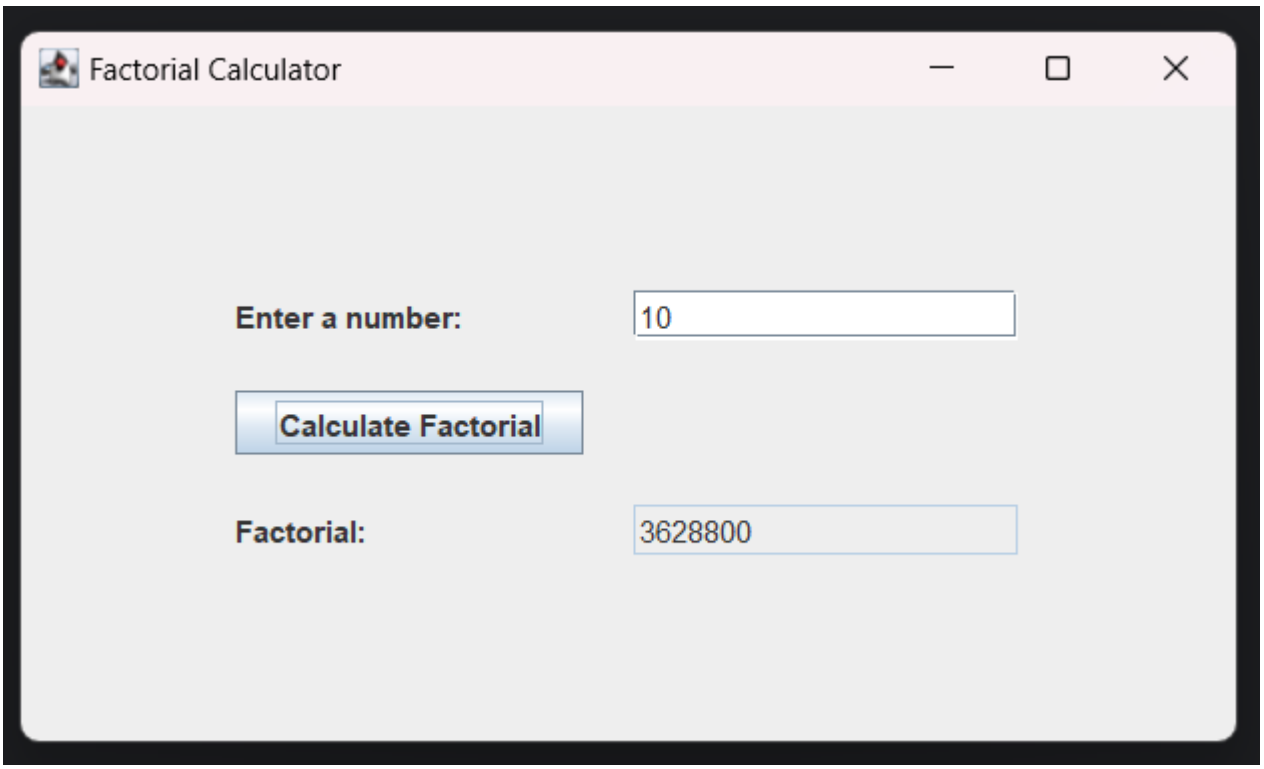
value of the text field is firstly converted into integer and then processed to find its factorial. The result will get display in another text field. (Hint: use swings)

Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Program1 extends JFrame {
   private JTextField inputField;
   private JTextField resultField;
   private JButton calculateButton;

   public Program1() {
      setTitle("Factorial Calculator");
      setSize(500, 500);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setLocationRelativeTo(null);

      JLabel inputLabel = new JLabel("Enter a number:");
      inputField = new JTextField(10);
      calculateButton = new JButton("Calculate Factorial");
      JLabel resultLabel = new JLabel("Factorial:");
      resultField = new JTextField(15);
      resultField.setEditable(false);

      setLayout(new GridBagLayout());

      GridBagConstraints gbc = new GridBagConstraints();
      gbc.fill = GridBagConstraints.HORIZONTAL;
      gbc.gridx = 0;
      gbc.gridy = 0;
      gbc.insets = new Insets(10, 10, 10, 10);
      add(inputLabel, gbc);

      gbc.gridx = 1;
      add(inputField, gbc);

      gbc.gridx = 0;
      gbc.gridy = 1;
      add(calculateButton, gbc);

      gbc.gridx = 0;
      gbc.gridy = 2;
      add(resultLabel, gbc);

      gbc.gridx = 1;
      add(resultField, gbc);

      calculateButton.addActionListener(new ActionListener() {
         @Override
         public void actionPerformed(ActionEvent e) {
```

```
            try {
                int number = Integer.parseInt(inputField.getText());
                long factorial = calculateFactorial(number);
                resultField.setText(String.valueOf(factorial));
            } catch (NumberFormatException ex) {
                resultField.setText("Invalid input");
            } catch (ArithmeticException ex) {
                resultField.setText("Overflow or negative input");
            }
        }
    });
}

private long calculateFactorial(int n) {
    if (n < 0) {
        throw new ArithmeticException("Factorial not defined for negative numbers");
    }
    if (n > 20) {
        throw new ArithmeticException("Number too large, will cause overflow");
    }

    long factorial = 1;
    for (int i = 1; i <= n; i++) {
        factorial *= i;
    }
    return factorial;
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new Program1().setVisible(true);
        }
    });
}
}
```

Output:



Program2: Write a java program that works as a simple calculator. Use a Grid Layout to arrange Buttons for digits and for the + - * % operations. Add a text field to display the result.

Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Program2 extends JFrame implements ActionListener {
    private JTextField displayField;
    private double firstNumber;
    private String operation;
    private boolean startNewInput;

    public Program2() {
        setTitle("Simple Calculator");
        setSize(400, 500);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        displayField = new JTextField();
        displayField.setPreferredSize(new Dimension(400, 50));
        displayField.setEditable(false);
        displayField.setHorizontalAlignment(JTextField.RIGHT);
        displayField.setFont(new Font("Arial", Font.BOLD, 24));

        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new GridLayout(4, 4, 5, 5));

        String[] buttonLabels = {
            "7", "8", "9", "+",
            "4", "5", "6", "-",
            "1", "2", "3", "*",
            "0", "C", "=", "%"
        };

        for (String label : buttonLabels) {
            JButton button = new JButton(label);
            button.addActionListener(this);
            button.setFont(new Font("Arial", Font.PLAIN, 18));
            buttonPanel.add(button);
        }

        setLayout(new BorderLayout(5, 5));
        add(displayField, BorderLayout.NORTH);
        add(buttonPanel, BorderLayout.CENTER);

        firstNumber = 0;
        operation = "";
        startNewInput = true;

        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
```

```
    String command = e.getActionCommand();

    if(command.matches("-")){
        if (startNewInput) {

            displayField.setText(command);
            startNewInput = false;
        } else {
            if(displayField.getText().equals("-")) {
                displayField.setText("");
                startNewInput = true;
            }
            else if (!displayField.getText().isEmpty()) {
            firstNumber = Double.parseDouble(displayField.getText());
            operation = command;
            startNewInput = true;
        }
        }
    }
    else if (command.matches("[0-9]")) {
        if (startNewInput) {
            displayField.setText(command);
            startNewInput = false;
        } else {
            displayField.setText(displayField.getText() + command);
        }
    }
    else if (command.matches("[+*%]")) {
        if (!displayField.getText().isEmpty()) {
            firstNumber = Double.parseDouble(displayField.getText());
            operation = command;
            startNewInput = true;
        }
    }
    else if (command.equals("=")) {
        if (!operation.isEmpty() && !displayField.getText().isEmpty()) {
            double secondNumber = Double.parseDouble(displayField.getText());
            double result = calculateResult(firstNumber, secondNumber, operation);
            displayField.setText(String.valueOf(result));
            operation = "";
            startNewInput = true;
        }
    }
    else if (command.equals("C")) {
        displayField.setText("");
        firstNumber = 0;
        operation = "";
        startNewInput = true;
    }
}

private double calculateResult(double n1, double n2, String op) {
    switch (op) {
```

```
            case "+":
                return n1 + n2;
            case "-":
                return n1 - n2;
            case "*":
                return n1 * n2;
            case "%":
                if (n2 == 0) {
                    JOptionPane.showMessageDialog(this, "Cannot divide by zero", "Error",
JOptionPane.ERROR_MESSAGE);
                    return 0;
                }
                return n1 % n2;
            default:
                return 0;
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new Program2());
    }
}
```

Output: