

1. Use Eclipse or Net bean platform and acquaint yourself with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.

```
import java.util.Scanner;

//import java.util.*;

public class Testp
{
    public static void main(String args[])
    {
        System.out.println("Welcome to JAVA Lab");
        System.out.println("Prime Number Program");
        Scanner sc= new Scanner(System.in);
        System.out.println("Enter valid Number");
        int n=sc.nextInt();
        int c=0;
        for(int i=1;i<=n;i++)
        {
            if(n%i==0)
            {
                c++;
            }
        }
        if(c==2)
        {
            System.out.println(n+" " +"is Prime Number");
        }
        else
        {
            System.out.println(n+" " +"is not Prime Number");
        }
    }
}
```

Output:

```
3is Prime Number
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Testp.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Testp
Welcome to JAVA Lab
Prime Number Program
Enter valid Number
1
1is not Prime Number
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Testp.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Testp
Welcome to JAVA Lab
Prime Number Program
Enter valid Number
4
4 is not Prime Number
```

2. Write a Java program to demonstrate the OOP principles. [i.e., Encapsulation, Inheritance, Polymorphism and Abstraction]

Inheritance

```
class Calculation{
    int z;
    public void addition(int x, int y){
        z=x+y;
        System.out.println("The sum of the given numbers:"+z);
    }
    public void Subtraction(int x, int y){
        z=x-y;
        System.out.println("The Difference between the given number:"+z);
    }
}

public class My_Calculation extends Calculation{
    public void multiplication(int x, int y){
        z=x*y;
        System.out.println("The product of the given number:"+z);
    }
    public static void main(String[] args){
        int a=20,b=10;
        My_Calculation demo = new My_Calculation();
        demo.addition(a, b);
    }
}
```

```

        demo.Subtraction(a, b);
        demo.multiplication(a, b);
    }
}

```

Output:

```

PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac My_Calculation.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java My_Calculation
The sum of the given numbers:30
The Difference between the given number:10
The product of the given number:200

```

Polymorphism(Compile Time Polymorphism)

```

class Overload{
    void demo(int a)
    {
        System.out.println("a:" +a);
    }
    void demo(int a, int b)
    {
        System.out.println("a and b:"+a +"," +b);
    }
    double demo(double a)
    {
        System.out.println("double a:" +a);
        return a*a;
    }
}

public class MethodOverloading {
    public static void main(String[] args)
    {
        Overload Obj = new Overload();
        double result;
        Obj.demo(10);
        Obj.demo(10,20);
        result = Obj.demo(5.5);
        System.out.println("O/P:" +result);
    }
}

```

```
    }  
}
```

Output:

```
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac MethodOverloading.java  
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java MethodOverloading  
a:10  
a and b:10,20  
double a:5.5  
O/P:30.25
```

Polymorphism(Run Time Polymorphism)

```
class Animal{  
    void place(){  
        System.out.println("Animals live on earth");  
    }  
}  
  
class Dog extends Animal{  
    void place(){  
        System.out.println("Dog lives in Kennel");  
    }  
}  
  
class Horse extends Animal{  
    void place(){  
        System.out.println(" Horse lives in stable");  
    }  
}  
  
class Rabbit extends Animal{  
    void place(){  
        System.out.println(" Rabbit lives in burrow");  
    }  
}  
  
public class Polymorphism {  
    public static void main( String args[]){  
        Animal A = new Animal();  
        A.place();  
        A = new Dog();
```

```

        A.place();
        A = new Horse();
        A.place();
        A = new Rabbit();
        A.place();
    }
}

```

Output:

```

PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Polymorphism.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Polymorphism
Animalslive on earth
Dog lives in Kennel
Horse lives in stable
Rabbit lives in burrow
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Polymorphism.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Polymorphism
Dog lives in Kennel

```

Abstraction

```

abstract class Sunstar{
    abstract void printInfo();
}

class Employee extends Sunstar{
    void printInfo(){
        String name = "Harjeeth";
        int age = 6;
        float salary = 000.00F;
        System.out.println(name);
        System.out.println(age);
        System.out.println(salary);
    }
}

public class Base {
    public static void main(String args[]){
        Sunstar s = new Employee();
        s.printInfo();
    }
}

```

Output:

```
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Base.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Base
Harjeeth
6
0.0
```

Encapsulation

```
class Person{
    private String name;
    private int age;
    public String getName(){
        return name;
    }
    public void setName(String name)
    {
        this.name = name;
    }
    public int getAge()
    {
        return age;
    }
    public void setAge(int age)
    {
        this.age = age;
    }
}

public class Encapsulation {
    public static void main(String args[])
    {
        Person person = new Person();
        person.setName("Sai");
        person.setAge(30);
        System.out.println("Name:" +person.getName() +"\n" +"Age:" + person.getAge());
    }
}
```

Output:

```
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Encapsulation.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Encapsulation
Name:Sai
Age:30
```

3. Write a Java program to handle checked and unchecked exceptions. Also, demonstrate the usage of custom exceptions in real time scenario.

Checked Exception

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
public class CheckedException {
    public static void main(String args[]){
        FileInputStream fis = null;
        try
        {
            fis = new FileInputStream("/home/techvidvan/file.txt");
        }
        catch(FileNotFoundException fnfe)
        {
            System.out.println("The specific file is not" +" " + "present at the given path");
        }
        int k;
        try{
            while((k = fis.read()) != -1)
            {
                System.out.println((char)k);
            }
            fis.close();
        }
        catch(IOException ioe)
        {
            System.out.println("I/O error occurred:" +ioe);
        }
    }
}
```

Output:

```
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac CheckedException.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java CheckedException
The specific file is not present at the given path
Exception in thread "main" java.lang.NullPointerException: Cannot invoke "java.io.FileInputStream.read()" because "<local1>" is null
    at CheckedException.main(CheckedException.java:18)
```

Unchecked Exception

```
public class Uncheckedexception {
    public static void main(String args[])
    {
        try{
            int arr[] = {1,2,3,4,5};
            System.out.println(arr[7]);
        }
        catch(ArrayIndexOutOfBoundsException e)
        {
            System.out.println("The specific index does not exist" + " " + "in array. Please correct the error");
        }
    }
}
```

Output:

```
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Uncheckedexception.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Uncheckedexception
The specific index does not exist in array. Please correct the error
```

Custom Checked Exception

```
class InvalidAgeException extends Exception{
    public InvalidAgeException(String str)
    {
        super(str);//calling the constructor of parent Exception
    }
}

public class CustomcheckedException {
    static void validate(int age) throws InvalidAgeException{
        if(age<18)
        {
            throw new InvalidAgeException("Age is not valid to Vote");
        }
    }
}
```



```

        else{
            System.out.println("Welcome to Vote");
        }
    }
}

public static void main(String[] args)
{
    try{
        validate(13);
    }
    catch(InvalidAgeException ex)
    {
        System.out.println("Caught the Exception");
        System.out.println("Exception occurred:" +ex);
    }
    System.out.println("rest of the code..");
}
}

```

Output:

```

PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac CustomcheckedException.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java CustomcheckedException
Caught the Exception
Exception occurred:InvalidAgeException: Age is not valid to Vote
rest of the code..

```

4. Write a Java program on Random Access File class to perform different read and write operations.

```

package javaProgram;

import java.io.RandomAccessFile;
import java.io.IOException;

public class RandomIO {

    public static void main( String args[]) throws IOException
    {
        RandomAccessFile file = new RandomAccessFile("myfile.dat","rw");
        file.writeChar('S');
        file.writeInt(2222);
        file.writeDouble(22.22);
        file.seek(0);///Moving the file pointer to the beginning
    }
}

```

```

        System.out.println(file.readChar());

        System.out.println(file.readInt());

        System.out.println(file.readDouble());

        file.seek(2);///Moving the file pointer to the 2nd item

        System.out.println(file.readInt());

        file.seek(file.length());

        file.writeBoolean(true);

        file.seek(4);///Moving the file pointer to the 4th item

        System.out.println(file.readBoolean());

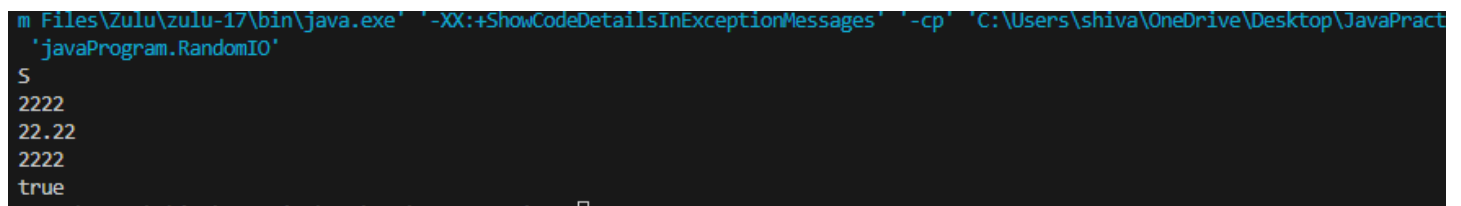
        file.close();

    }

}

```

Output:



```

m Files\Zulu\zulu-17\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\shiva\OneDrive\Desktop\JavaPract'
'javaProgram.RandomIO'
5
2222
22.22
2222
true

```

5. Write a Java program to demonstrate the working of different collection classes. [Use package structure to store multiple classes].

Array List with a programmatic

```

import java.util.ArrayList;

import java.util.Iterator;

public class ArrayListExample {

    // @SuppressWarnings("unchecked")

    public static void main(String args[])

    {

        // @SuppressWarnings("rawtypes")

        ArrayList al = new ArrayList();

        al.add("Hai");

        al.add("Hello");

        // @SuppressWarnings("rawtypes")

        Iterator itr = al.iterator();

        while(itr.hasNext())

        {

            System.out.println(itr.next());

        }

    }

}

```

```
    }  
    }  
}
```

Output:

```
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac ArrayListExample.java  
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java ArrayListExample  
Hai  
Hello
```

Linked List with a Programmatic

```
import java.util.LinkedList;  
import java.util.Iterator;  
public class LinkedListExample {  
    public static void main(String[] args)  
    {  
        LinkedList<String> ll = new LinkedList<String>();  
        ll.add("Sree");  
        ll.add("Vij");  
        ll.add("shiv");  
        ll.add("Har");  
        Iterator<String> itr = ll.iterator();  
        while(itr.hasNext())  
        {  
            System.out.println(itr.next());  
        }  
    }  
}
```

Output:

```
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac LinkedListExample.java  
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java LinkedListExample  
Sree  
Vij  
shiv  
Har
```

Priority queues with a programmatic

```
import java.util.Iterator;
import java.util.PriorityQueue;
public class QueueExample {
    public static void main(String args[]) {
        PriorityQueue<String> queue = new PriorityQueue<String>();
        queue.add("Sree");
        queue.add("Vij");
        queue.add("Shiva");
        queue.add("Hari");
        System.out.println("head:" +queue.element());
        System.out.println("head:" +queue.peek());
        System.out.println("iterating the queue elements:");
        // @SuppressWarnings("rawtypes")
        Iterator itr = queue.iterator();
        while (itr.hasNext()) {
            System.out.println(itr.next());
        }
        queue.remove();
        queue.poll();
        System.out.println("after removing two elements:");
        Iterator<String> itr2 = queue.iterator();
        while (itr2.hasNext()) {
            System.out.println(itr2.next());
        }
    }
}
```

Output:

```

PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac QueueExample.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java QueueExample
head:Hari
head:Hari
iterating the queue elements:
Hari
Shiva
Sree
Vij
after removing two elements:
Sree
Vij

```

Vector with a programmatic

```

import java.util.Iterator;

import java.util.Vector;

public class Vectorcollection {

    public static void main(String[] args)

    {

        Vector<String> v = new Vector<String>();

        v.add("Sree");

        v.add("Vije");

        v.add("Shiva");

        v.add("Hari");

        Iterator<String> itr = v.iterator();

        while (itr.hasNext()) {

            System.out.println(itr.next());

        }

    }

}

```

Output:

```

PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Vectorcollection.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Vectorcollection
Sree
Vije
Shiva
Hari

```

Stack with a programmatic

```

import java.util.Iterator;

import java.util.Stack;

public class StackCollection {

    public static void main(String[] args)

```

```

{
    Stack<String> stack = new Stack<String>();
    stack.push("Sree");
    stack.push("Vije");
    stack.push("Shiv");
    stack.push("Hari");
    stack.pop();
    Iterator<String> itr = stack.iterator();
    while(itr.hasNext())
    {
        System.out.println(itr.next());
    }
}
}

```

Output:

```

PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac StackCollection.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java StackCollection
Sree
Vije
Shiv

```

6. Write a program to synchronize the threads acting on the same object. [Consider the example of any reservations like railway, bus, movie ticket booking, etc.]

```

class Reserve extends Thread{
    int available = 1;
    int wanted;
    public Reserve(int i)
    {
        wanted = i;
    }
    public void run()
    {
        System.out.println("Available berths="+available);
        if(available >= wanted)
        {
            String name = Thread.currentThread().getName();
            System.out.println(wanted+"Berth(s) reserved for" +name);
            try{

```

```

        Thread.sleep(1500);
        available = available-wanted;
    }
    catch(InterruptedException ie)
    {
        ie.printStackTrace();
    }
}
else{
    System.out.println("Sorry! No berths available");
}
}
}

public class Threadsprog {
    public static void main(String args[])
    {
        Reserve obj = new Reserve(1);
        Thread t1 = new Thread(obj);
        Thread t2 = new Thread(obj);
        t1.setName("FirstPerson");
        t2.setName("Second Person");
        t1.start();
        t2.start();
    }
}

```

Output:

```

PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Threadsprog.java
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Threadsprog
Available berths=1
Available berths=1
1Berth(s) reserved forFirstPerson
1Berth(s) reserved forSecond Person

```

7. Write a program to perform CRUD operations on the student table in a database using JDBC.

JDBC Connection Program

```

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

```

```

public class JDBCConnection {
    public static void main(String[] args){
        try{
            Connection conn = getConnection();
        }
        catch(SQLException e){
            e.printStackTrace();
        }
    }

    public static Connection getConnection() throws SQLException{
        String url = "jdbc:mysql://localhost:3306/DatabaseName";
        String user = "userid";
        String password = "password";
        Connection conn = DriverManager.getConnection(url, user, password);
        System.out.println("Successfully Connected");
        return conn;
    }
}

```

Output:

Complete Program

```

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class CRUD {
    public static void main(String[] args) throws SQLException{
        String url = "jdbc:mysql://localhost:3306/TestDatabase";
        String user = "userid";
        String password = "password";
        Connection conn = DriverManager.getConnection(url, user, password);
        System.out.println("Successfully Connected");
        Statement stmt = conn.createStatement();
    }
}

```



```

        int rows = stmt.executeUpdate("insert into employee(age,name)
values(23,'James')");

        System.out.println("Rows inserted="+rows);

        rows = stmt.executeUpdate("Update employee set age=31 where
name='James'");

        System.out.println("Rows updated=" +rows);

        ResultSet rs = stmt.executeQuery("Select * from employee");
        while(rs.next())
        {
            System.out.println("Emp Id:"
+rs.getInt("id")+",Name:"+rs.getString("name")+",Age:"+rs.getInt("age"));
        }

        rows=stmt.executeUpdate("delete from employee where name='James'");

        System.out.println("Rows deleted="+rows);
    }
}

```

Output:

8. Write a Java program that works as a simple calculator. Use a grid layout to arrange buttons for the digits and for the +, -, *, % operations. Add a text field to display the result. Handle any possible exceptions like divided by zero.

```

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.*;

class A extends JFrame implements ActionListener{

    public JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,b10,b11,b12,b13,b14,b15,b16;
    public JTextField tf1;
    public JPanel p;
    public String v="";
    public String v1="0";
    public String op="";

    public A()

```

```
{  
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    setSize(400,400);  
    p = new JPanel(new FlowLayout());  
    tf1 = new JTextField(10);  
    p.add(tf1);  
    add(p);  
    setLayout(new GridLayout(0,3));  
    b1 = new JButton("1");  
    b1.addActionListener(this);  
    add(b1);  
    b2 = new JButton("2");  
    b2.addActionListener(this);  
    add(b2);  
    b3 = new JButton("3");  
    b3.addActionListener(this);  
    add(b3);  
    b4 = new JButton("4");  
    b4.addActionListener(this);  
    add(b4);  
    b5 = new JButton("5");  
    b5.addActionListener(this);  
    add(b5);  
    b6 = new JButton("6");  
    b6.addActionListener(this);  
    add(b6);  
    b7 = new JButton("7");  
    b7.addActionListener(this);  
    add(b7);  
    b8 = new JButton("8");  
    b8.addActionListener(this);  
    add(b8);  
    b9 = new JButton("9");  
    b9.addActionListener(this);  
    add(b9);  
}
```

```

        b10 = new JButton("0");
        b10.addActionListener(this);
        add(b10);
        b11 = new JButton("+");
        b11.addActionListener(this);
        add(b11);
        b12 = new JButton("-");
        b12.addActionListener(this);
        add(b12);
        b13 = new JButton("*");
        b13.addActionListener(this);
        add(b13);
        b14 = new JButton("/");
        b14.addActionListener(this);
        add(b14);
        b15 = new JButton("=");
        b15.addActionListener(this);
        add(b15);
        b16 = new JButton("%");
        b16.addActionListener(this);
        add(b16);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent ae){
        String b = ae.getActionCommand();
        switch(b){
            case "1":{
                v = v + "1";
                tf1.setText(v);
            }
            break;
            case "2":{
                v = v + "2";
                tf1.setText(v);
            }
        }
    }

```

```
break;
case "3":{
    v = v + "3";
    tf1.setText(v);
}
break;
case "4":{
    v = v + "4";
    tf1.setText(v);
}
break;
case "5":{
    v = v + "5";
    tf1.setText(v);
}
break;
case "6":{
    v = v + "6";
    tf1.setText(v);
}
break;
case "7":{
    v = v + "7";
    tf1.setText(v);
}
break;
case "8":{
    v = v + "8";
    tf1.setText(v);
}
break;
case "9":{
    v = v + "9";
    tf1.setText(v);
}
```

```
break;
case "0":{
    v = v + "0";
    tf1.setText(v);
}
break;
case "+":{
    op = "+";
    v1=tf1.getText();
    v="";
}
break;
case "-":{
    op = "-";
    v1=tf1.getText();
    v="";
}
break;
case "*":{
    op = "*";
    v1=tf1.getText();
    v="";
}
break;
case "/":{
    op = "/";
    v1=tf1.getText();
    v="";
}
break;
case "%":{
    op = "%";
    v1=tf1.getText();
    v="";
}
```

```

break;
case "=": {
    switch(op) {
        case "+": {
            v = tf1.getText();
            if(v.equals("")) {
                v="0";
            }
            long i = Long.parseLong(v1)+Long.parseLong(v);
            tf1.setText(String.valueOf(i));
            v="";
        }
        break;
        case "-": {
            v = tf1.getText();
            if(v.equals("")) {
                v="0";
            }
            long i = Long.parseLong(v1)-Long.parseLong(v);
            tf1.setText(String.valueOf(i));
            v="";
        }
        break;
        case "*": {
            v = tf1.getText();
            if(v.equals("")) {
                v="0";
            }
            long i = Long.parseLong(v1)*Long.parseLong(v);
            tf1.setText(String.valueOf(i));
            v="";
        }
        break;
        case "/": {
            try {

```

```

        v = tf1.getText();
        if(v.equals("")){
            v="0";
        }
        long i = Long.parseLong(v1)/Long.parseLong(v);
        tf1.setText(String.valueOf(i));
        v="";
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(this, ex.getMessage());
    }
}
break;
case "%":{
    try{
        v = tf1.getText();
        if(v.equals("")){
            v="0";
        }
        long i = Long.parseLong(v1)%Long.parseLong(v);
        tf1.setText(String.valueOf(i));
        v="";
    }
    catch(Exception ex)
    {
        JOptionPane.showMessageDialog(this, ex.getMessage());
    }
}
break;
}
}
break;
}
}

```

```

    }

    public class Calc {

        public static void main(String args[]){

            A a = new A();

        }

    }

```

Output:

The screenshot shows an IDE with a Java file named `Calc.java` and a running application window titled `Calc`. The code in the IDE includes:

```

12 public JTextField tf1;
13 public JPanel p;
14 public String v="";
15 public String v1="0";
16 public String op="";
17 public A()

```

The IDE's `PROBLEMS` tab shows two errors:

```

Calc.java:66: error: incompatible types: A cannot be converted to Action
    b15.addActionListener(this);
    ^
Calc.java:69: error: incompatible types: A cannot be converted to Action
    b16.addActionListener(this);
    ^

```

The application window displays a calculator interface with a text input field and buttons for digits 0-9, operators (+, -, *, /, =, %), and a clear button.

9. Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired. [Use Adapter classes]

```

import java.awt.Color;

import java.awt.FlowLayout;

import java.awt.Font;

import java.awt.event.MouseEvent;

import java.awt.event.MouseListener;

import javax.swing.JFrame;

import javax.swing.JLabel;

public class MouseEvents extends JFrame implements MouseListener {

    JLabel JL;

    public MouseEvents()

    {

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setSize(300, 300);

```



```

        setLayout(new FlowLayout(FlowLayout.CENTER));
        JL = new JLabel();
        Font f = new Font("Verdana", Font.BOLD, 20);
        JL.setFont(f);
        JL.setForeground(Color.BLUE);
        add(JL);
        addMouseListener(this);
        setVisible(true);
    }

    public void mouseExited(MouseEvent m)
    {
        JL.setText("Mouse Exited");
    }

    public void mouseEntered(MouseEvent m)
    {
        JL.setText("Mouse Entered");
    }

    public void mouseReleased(MouseEvent m)
    {
        JL.setText("Mouse Released");
    }

    public void mousePressed(MouseEvent m)
    {
        JL.setText("Mouse Pressed");
    }

    public void mouseClicked(MouseEvent m)
    {
        JL.setText("Mouse Clicked");
    }

    public static void main(String[] args){
        MouseEvents me = new MouseEvents();
    }
}

```

Output:

```
35 | }  
PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS  
Calc.java:66: error: incompatible types: A cannot be converted to ActionListener  
    b15.addActionListener(this);  
                        ^  
Calc.java:69: error: incompatible types: A cannot be converted to ActionListener  
    b16.addActionListener(this);  
                        ^  
Note: Some messages have been simplified; recompile with -Xdiags:verbose to get  
19 errors  
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac Calc.java  
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Calc  
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java Calc  
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> javac MouseEvents.java  
PS C:\Users\shiva\OneDrive\Desktop\JavaPractice\src> java MouseEvents  
[]
```

