**Assignment 1 :**

1. Create a Hash table containing Employee name and Salary. Display the details of the hash table. Also search for a specific Employee and display Salary of that Employee.

import java.util.HashMap;

import java.util.Scanner;

public class EmployeeHashTable {

public static void main(String[] args) {

HashMap<String, Double> employeeTable = new HashMap();

employeeTable.put("Alice", 70000.0);

employeeTable.put("Bob", 85000.0);

employeeTable.put("Charlie", 65000.0);

employeeTable.put("Diana", 90000.0);

employeeTable.put("Ethan", 75000.0);

System.out.println("Employee Details:");

for (String name : employeeTable.keySet()) {

System.out.println("Name: " + name + ", Salary: $" + employeeTable.get(name));

}

Scanner scanner = new Scanner(System.in);

System.out.print("\nEnter the employee name to search: ");

String searchName = scanner.nextLine();

if (employeeTable.containsKey(searchName)) {

System.out.println(searchName + "'s Salary is: $" + employeeTable.get(searchName));

} else {

System.out.println("Employee named '" + searchName + "' not found.");

}

scanner.close();

}

}

1. Write a Java program to accept the details of students (rno, sname, per) at least 5 Records, store it into database and display the details

import java.sql.\*;

import java.util.Scanner;

public class StudentDatabase {

public static void main(String[] args) {

Class.forName(Give the Driver Name );

String url = "jdbc:Postgresql::students.db"; // Write correct URL

try (Connection conn = DriverManager.getConnection(url)) {

if (conn != null) {

createTable(conn); // Create table if not exists

insertStudents(conn); // Insert at least 5 student records

displayStudents(conn); // Display all records

}

} catch (SQLException e) {

System.out.println(e.getMessage());

}

}

public static void createTable(Connection conn) throws SQLException {

String createTableSQL = """

CREATE TABLE IF NOT EXISTS students (

rno INTEGER PRIMARY KEY,

sname TEXT NOT NULL,

per REAL NOT NULL

);

""";

try (Statement stmt = conn.createStatement()) {

stmt.execute(createTableSQL);

}

}

public static void insertStudents(Connection conn) throws SQLException {

Scanner sc = new Scanner(System.in);

String insertSQL = "INSERT OR IGNORE INTO students(rno, sname, per) VALUES (?, ?, ?)";

try (PreparedStatement pstmt = conn.prepareStatement(insertSQL)) {

int count = 0;

while (count < 5) {

System.out.println("Enter details for student " + (count + 1));

System.out.print("Roll Number: ");

int rno = sc.nextInt();

sc.nextLine(); // consume newline

System.out.print("Student Name: ");

String sname = sc.nextLine();

System.out.print("Percentage: ");

double per = sc.nextDouble();

sc.nextLine(); // consume newline

// Set parameters and execute insert

pstmt.setInt(1, rno);

pstmt.setString(2, sname);

pstmt.setDouble(3, per);

pstmt.executeUpdate();

count++;

System.out.println("Student record added!\n");

}

}

}

public static void displayStudents(Connection conn) throws SQLException {

String selectSQL = "SELECT \* FROM students";

try (Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(selectSQL)) {

System.out.println("Student Details:");

System.out.println("-------------------------------");

System.out.printf("%-10s %-20s %-10s\n", "Roll No", "Name", "Percentage");

while (rs.next()) {

int rno = rs.getInt("rno");

String sname = rs.getString("sname");

double per = rs.getDouble("per");

System.out.printf("%-10d %-20s %-10.2f\n", rno, sname, per);

}

}

}

}

**Assignment 2:**

1. Accept ‘n’ integers from the user. Store and display integers in sorted order having proper collection class. The collection should not accept duplicate elements.

import java.util.Scanner;

import java.util.TreeSet;

public class SortedUniqueIntegers {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

TreeSet<Integer> numbers = new TreeSet<>();

System.out.print("Enter how many integers you want to input: ");

int n = sc.nextInt();

System.out.println("Enter " + n + " integers:");

for (int i = 0; i < n; i++) {

int num = sc.nextInt();

boolean added = numbers.add(num);

if (!added) {

System.out.println(num + " is a duplicate and will not be added.");

}

}

System.out.println("\nSorted unique integers are:");

for (int num : numbers) {

System.out.print(num + " ");

}

sc.close();

}

}

1. ­Write a java program to design a following GUI. Use appropriate Layout and Components.

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class LoginForm {

public static void main(String[] args) {

JFrame frame = new JFrame("Login Form");

frame.setSize(350, 200);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(new GridLayout(3, 2, 10, 10));

JLabel lblUsername = new JLabel("Username:");

JTextField txtUsername = new JTextField();

JLabel lblPassword = new JLabel("Password:");

JPasswordField txtPassword = new JPasswordField();

JButton btnLogin = new JButton("Login");

JButton btnReset = new JButton("Reset");

btnReset.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

txtUsername.setText("");

txtPassword.setText("");

}

});

btnLogin.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

JOptionPane.showMessageDialog(frame, "Login Successful!");

}

});

frame.add(lblUsername);

frame.add(txtUsername);

frame.add(lblPassword);

frame.add(txtPassword);

frame.add(btnLogin);

frame.add(btnReset);

frame.setLocationRelativeTo(null);

frame.setVisible(true);

}

}

**Assignment 3:**

1. Write a Java program to accept the details of Student (RNo, SName, Per, Gender, Class) and store into the database. (Use appropriate Swing Components and PreparedStatement Interface)

import java.sql.\*;

import java.util.Scanner;

public class StudentConsoleDB {

Class.forName() ;

private static final String DB\_URL = "jdbc:\_\_\_\_\_\_\_\_\_:students.db";

public static void main(String[] args) {

createTableIfNotExists();

try (Scanner sc = new Scanner(System.in)) {

System.out.println("Enter details of 5 students:");

for (int i = 1; i <= 5; i++) {

System.out.println("\nStudent " + i + ":");

System.out.print("Roll Number (int): ");

int rno = sc.nextInt();

sc.nextLine(); // consume newline

System.out.print("Student Name: ");

String sname = sc.nextLine();

System.out.print("Percentage (double): ");

double per = sc.nextDouble();

sc.nextLine();

System.out.print("Gender (Male/Female/Other): ");

String gender = sc.nextLine();

System.out.print("Class: ");

String className = sc.nextLine();

insertStudent(rno, sname, per, gender, className);

}

System.out.println("\nAll students added successfully!\n");

displayAllStudents();

} catch (Exception e) {

System.out.println("Error: " + e.getMessage());

}

}

private static void createTableIfNotExists() {

String createTableSQL = """

CREATE TABLE IF NOT EXISTS students (

rno INTEGER PRIMARY KEY,

sname TEXT NOT NULL,

per REAL NOT NULL,

gender TEXT NOT NULL,

class TEXT NOT NULL

);

""";

try (Connection conn = DriverManager.getConnection(DB\_URL);

Statement stmt = conn.createStatement()) {

stmt.execute(createTableSQL);

} catch (SQLException e) {

System.out.println("DB Error (creating table): " + e.getMessage());

}

}

private static void insertStudent(int rno, String sname, double per, String gender, String className) {

String insertSQL = "INSERT INTO students (rno, sname, per, gender, class) VALUES (?, ?, ?, ?, ?)";

try (Connection conn = DriverManager.getConnection(DB\_URL);

PreparedStatement pstmt = conn.prepareStatement(insertSQL)) {

pstmt.setInt(1, rno);

pstmt.setString(2, sname);

pstmt.setDouble(3, per);

pstmt.setString(4, gender);

pstmt.setString(5, className);

pstmt.executeUpdate();

} catch (SQLException e) {

System.out.println("DB Error (inserting student): " + e.getMessage());

}

}

private static void displayAllStudents() {

String selectSQL = "SELECT \* FROM students";

try (Connection conn = DriverManager.getConnection(DB\_URL);

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(selectSQL)) {

System.out.println("RNo\tName\tPercentage\tGender\tClass");

System.out.println("--------------------------------------------------");

while (rs.next()) {

int rno = rs.getInt("rno");

String sname = rs.getString("sname");

double per = rs.getDouble("per");

String gender = rs.getString("gender");

String className = rs.getString("class");

System.out.printf("%d\t%s\t%.2f\t\t%s\t%s\n", rno, sname, per, gender, className);

}

} catch (SQLException e) {

System.out.println("DB Error (fetching data): " + e.getMessage());

}

}

}

**Assignment 4:**

1. Write Java program to demonstrate any five different HashSet operations.

import java.util.HashSet;

public class HashSetDemo {

public static void main(String[] args) {

HashSet<String> set = new HashSet<>();

set.add("Apple");

set.add("Banana");

set.add("Cherry");

set.add("Date");

set.add("Elderberry");

System.out.println("Initial HashSet: " + set);

String checkFruit = "Banana";

System.out.println("Does the set contain '" + checkFruit + "'? " + set.contains(checkFruit));

set.remove("Date");

System.out.println("After removing 'Date': " + set);

System.out.print("Iterating over HashSet elements: ");

for (String fruit : set) {

System.out.print(fruit + " ");

}

System.out.println();

set.clear();

System.out.println("After clearing, is the set empty? " + set.isEmpty());

}

}

1. Write a JDBC program to count number of records in a Emp table.

import java.sql.\*;

public class EmpRecordCount {

private static final String DB\_URL = "jdbc:mysql://localhost:3306/your\_database"; // Example for MySQL

private static final String USER = "your\_username";

private static final String PASS = "your\_password";

public static void main(String[] args) {

String query = "SELECT COUNT(\*) AS total FROM Emp";

try (

// Establish connection

Connection conn = DriverManager.getConnection(DB\_URL, USER, PASS);

// Create statement

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(query);

) {

if (rs.next()) {

int count = rs.getInt("total");

System.out.println("Total records in Emp table: " + count);

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**Assignment 5:**

1. Write Java program to demonstrate any five different LinkedList operations

import java.util.LinkedList;

public class LinkedListDemo {

public static void main(String[] args) {

LinkedList<String> list = new LinkedList<>();

list.add("Red");

list.add("Green");

list.add("Blue");

list.add("Yellow");

list.add("Purple");

System.out.println("Initial LinkedList: " + list);

list.addFirst("Black");

System.out.println("After addFirst('Black'): " + list);

list.remove("Green");

System.out.println("After removing 'Green': " + list);

System.out.println("Element at index 2: " + list.get(2));

System.out.print("Iterating over LinkedList: ");

for (String color : list) {

System.out.print(color + " ");

}

System.out.println();

}

}

1. Write a java program to design a following GUI. Use appropriate Layout and Components.

import java.awt.event.\*;

import javax.swing.\*;

public class Vaccine{

public static void main(String[] args) {

JFrame f = new JFrame("Vaccination");

f.setSize(400, 300);

f.setLayout(null);

f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JLabel l1 = new JLabel("Vaccination Details");

l1.setBounds(30, 10, 150, 20);

JLabel l2 = new JLabel("Name:");

l2.setBounds(30, 40, 50, 20);

JTextField t1 = new JTextField();

t1.setBounds(90, 40, 200, 20);

f.add(l1); f.add(l2); f.add(t1);

JCheckBox d1 = new JCheckBox("1st Dose");

d1.setBounds(30, 80, 100, 20);

JCheckBox d2 = new JCheckBox("2nd Dose");

d2.setBounds(30, 100, 100, 20);

f.add(d1); f.add(d2);

JRadioButton v1 = new JRadioButton("Covishield");

v1.setBounds(200, 80, 100, 20);

JRadioButton v2 = new JRadioButton("Covaxin");

v2.setBounds(200, 100, 100, 20);

JRadioButton v3 = new JRadioButton("Sputnik V");

v3.setBounds(200, 120, 100, 20);

ButtonGroup bg = new ButtonGroup();

bg.add(v1); bg.add(v2); bg.add(v3);

f.add(v1); f.add(v2); f.add(v3);

JLabel o1 = new JLabel();

o1.setBounds(30, 170, 350, 20);

JLabel o2 = new JLabel();

o2.setBounds(30, 190, 350, 20);

f.add(o1); f.add(o2);

JButton b = new JButton("Show");

b.setBounds(80, 150, 100, 25);

f.add(b);

b.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String name = t1.getText();

String dose1 = d1.isSelected() ? "Yes" : "No";

String dose2 = d2.isSelected() ? "Yes" : "No";

String vac = "";

if (v1.isSelected()) vac = "Covishield";

else if (v2.isSelected()) vac = "Covaxin";

else if (v3.isSelected()) vac = "Sputnik V";

o1.setText("Name: " + name + " 1st dose: " + dose1 + " 2nd dose: " + dose2);

o2.setText("Vaccine: " + vac);

}

});

f.setVisible(true);

}

**Assignment 6:**

1. Write Java program to demonstrate any five different ArrayList operations

import java.util.ArrayList;

public class ArrayListDemo {

public static void main(String[] args) {

ArrayList<String> list = new ArrayList<>();

list.add("Java");

list.add("Python");

list.add("C++");

list.add("JavaScript");

list.add("Ruby");

System.out.println("Initial ArrayList: " + list);

list.add(2, "C#");

System.out.println("After adding 'C#' at index 2: " + list);

list.remove("Ruby");

System.out.println("After removing 'Ruby': " + list);

list.set(3, "TypeScript");

System.out.println("After updating element at index 3 to 'TypeScript': " + list);

System.out.print("Iterating over ArrayList: ");

for (String lang : list) {

System.out.print(lang + " ");

}

System.out.println();

}

}

1. Write a menu driven program in Java for the following: Assume Emp table with attributes ( ENo, EName, salary, Desg ) is already created. 1. Insert 2. Display 3. Exit.

import java.sql.\*;

import java.util.Scanner;

public class EmpMenuDriven {

private static final String DB\_URL = "jdbc:sqlite:emp.db"; // Change for your DB

// For MySQL, for example:

// private static final String DB\_URL = "jdbc:mysql://localhost:3306/your\_db";

// private static final String USER = "your\_user";

// private static final String PASS = "your\_pass";

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int choice;

while (true) {

System.out.println("\n===== EMPLOYEE MENU =====");

System.out.println("1. Insert Employee");

System.out.println("2. Display Employees");

System.out.println("3. Exit");

System.out.print("Enter your choice: ");

choice = sc.nextInt();

sc.nextLine(); // consume newline

switch (choice) {

case 1:

insertEmployee(sc);

break;

case 2:

displayEmployees();

break;

case 3:

System.out.println("Exiting program...");

sc.close();

System.exit(0);

break;

default:

System.out.println("Invalid choice. Please try again.");

}

}

}

private static void insertEmployee(Scanner sc) {

try (Connection conn = DriverManager.getConnection(DB\_URL)) {

System.out.print("Enter Employee Number (ENo): ");

int eno = sc.nextInt();

sc.nextLine(); // consume newline

System.out.print("Enter Employee Name (EName): ");

String ename = sc.nextLine();

System.out.print("Enter Salary: ");

double salary = sc.nextDouble();

sc.nextLine();

System.out.print("Enter Designation (Desg): ");

String desg = sc.nextLine();

String insertSQL = "INSERT INTO Emp (ENo, EName, salary, Desg) VALUES (?, ?, ?, ?)";

try (PreparedStatement pstmt = conn.prepareStatement(insertSQL)) {

pstmt.setInt(1, eno);

pstmt.setString(2, ename);

pstmt.setDouble(3, salary);

pstmt.setString(4, desg);

int rows = pstmt.executeUpdate();

if (rows > 0) {

System.out.println("Employee inserted successfully!");

} else {

System.out.println("Failed to insert employee.");

}

}

} catch (SQLException e) {

System.out.println("DB Error: " + e.getMessage());

}

}

private static void displayEmployees() {

try (Connection conn = DriverManager.getConnection(DB\_URL);

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM Emp")) {

System.out.println("\nENo\tEName\tSalary\t\tDesg");

System.out.println("---------------------------------------------");

boolean empty = true;

while (rs.next()) {

empty = false;

int eno = rs.getInt("ENo");

String ename = rs.getString("EName");

double salary = rs.getDouble("salary");

String desg = rs.getString("Desg");

System.out.printf("%d\t%s\t%.2f\t\t%s\n", eno, ename, salary, desg);

}

if (empty) {

System.out.println("No employee records found.");

}

} catch (SQLException e) {

System.out.println("DB Error: " + e.getMessage());

}

}

}

**Assignment 7:**

1. Write a menu driven program in Java for the following: Assume Emp table with attributes ( ENo, EName, salary, Desg ) is already created. 1. Update 2. Display 2. Exit.

import java.sql.\*;

import java.util.Scanner;

public class EmpMenuDrivenUpdateDisplay {

private static final String DB\_URL = "jdbc:sqlite:emp.db";

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int choice;

while (true) {

System.out.println("\n===== EMPLOYEE MENU =====");

System.out.println("1. Update Employee");

System.out.println("2. Display Employees");

System.out.println("3. Exit");

System.out.print("Enter your choice: ");

choice = sc.nextInt();

sc.nextLine(); // consume newline

switch (choice) {

case 1:

updateEmployee(sc);

break;

case 2:

displayEmployees();

break;

case 3:

System.out.println("Exiting program...");

sc.close();

System.exit(0);

break;

default:

System.out.println("Invalid choice. Please try again.");

}

}

}

private static void updateEmployee(Scanner sc) {

System.out.print("Enter Employee Number (ENo) to update: ");

int eno = sc.nextInt();

sc.nextLine();

System.out.println("What do you want to update?");

System.out.println("1. Employee Name");

System.out.println("2. Salary");

System.out.println("3. Designation");

System.out.print("Enter your choice: ");

int updateChoice = sc.nextInt();

sc.nextLine();

String column = null;

Object newValue = null;

switch (updateChoice) {

case 1:

System.out.print("Enter new Employee Name: ");

newValue = sc.nextLine();

column = "EName";

break;

case 2:

System.out.print("Enter new Salary: ");

newValue = sc.nextDouble();

sc.nextLine();

column = "salary";

break;

case 3:

System.out.print("Enter new Designation: ");

newValue = sc.nextLine();

column = "Desg";

break;

default:

System.out.println("Invalid update choice.");

return;

}

String updateSQL = "UPDATE Emp SET " + column + " = ? WHERE ENo = ?";

try (Connection conn = DriverManager.getConnection(DB\_URL);

PreparedStatement pstmt = conn.prepareStatement(updateSQL)) {

if (newValue instanceof String) {

pstmt.setString(1, (String) newValue);

**Assignment 8:**

1. Write Java program to demonstrate any five different LinkedHashSet operations

import java.util.LinkedHashSet;

public class LinkedHashSetDemo {

public static void main(String[] args) {

LinkedHashSet<String> set = new LinkedHashSet<>();

set.add("Apple");

set.add("Banana");

set.add("Cherry");

set.add("Date");

set.add("Elderberry");

System.out.println("Initial LinkedHashSet: " + set);

System.out.println("Does set contain 'Cherry'? " + set.contains("Cherry"));

set.remove("Date");

System.out.println("After removing 'Date': " + set);

System.out.print("Iterating over LinkedHashSet elements: ");

for (String fruit : set) {

System.out.print(fruit + " ");

}

System.out.println();

set.clear();

System.out.println("After clearing, is the set empty? " + set.isEmpty());

}

}

1. Write a JSP program to check whether a given number is even or not.

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<title>Check Even or Odd</title>

</head>

<body>

<h2>Check if a Number is Even or Odd</h2>

<form method="post" action="">

Enter a number: <input type="number" name="number" required />

<input type="submit" value="Check" />

</form>

<%

String numStr = request.getParameter("number");

if (numStr != null) {

try {

int number = Integer.parseInt(numStr);

if (number % 2 == 0) {

out.println("<h3>" + number + " is Even.</h3>");

} else {

out.println("<h3>" + number + " is Odd.</h3>");

}

} catch (NumberFormatException e) {

out.println("<h3>Invalid input. Please enter a valid integer.</h3>");

}

}

%>

</body>

</html>

**Assignment 9 :**

1. Write Java program to demonstrate any five different TreeSet operations

import java.util.TreeSet;

public class TreeSetDemo {

public static void main(String[] args) {

TreeSet<String> fruits = new TreeSet<>();

fruits.add("Banana");

fruits.add("Apple");

fruits.add("Mango");

fruits.add("Orange");

fruits.add("Grapes");

System.out.println("Original TreeSet: " + fruits);

fruits.remove("Mango");

System.out.println("After removing 'Mango': " + fruits);

if (fruits.contains("Apple")) {

System.out.println("'Apple' is present in the TreeSet.");

} else {

System.out.println("'Apple' is not present in the TreeSet.");

}

System.out.println("Iterating over TreeSet:");

for (String fruit : fruits) {

System.out.println(fruit);

}

System.out.println("First element: " + fruits.first());

System.out.println("Last element: " + fruits.last());

}

}

1. Write a java program to implement a simple arithmetic calculator using AWT / Swing.

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class CalculatorSwing extends JFrame implements ActionListener {

JTextField num1Field, num2Field;

JLabel resultLabel;

JButton addBtn, subBtn, mulBtn, divBtn;

CalculatorSwing() {

setTitle("Swing Calculator");

num1Field = new JTextField(10);

num2Field = new JTextField(10);

resultLabel = new JLabel("Result: ");

addBtn = new JButton("+");

subBtn = new JButton("-");

mulBtn = new JButton("\*");

divBtn = new JButton("/");

setLayout(new FlowLayout());

add(new JLabel("Number 1:"));

add(num1Field);

add(new JLabel("Number 2:"));

add(num2Field);

add(addBtn);

add(subBtn);

add(mulBtn);

add(divBtn);

add(resultLabel);

addBtn.addActionListener(this);

subBtn.addActionListener(this);

mulBtn.addActionListener(this);

divBtn.addActionListener(this);

setSize(300, 200);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent e) {

try {

double num1 = Double.parseDouble(num1Field.getText());

double num2 = Double.parseDouble(num2Field.getText());

double result = 0;

if (e.getSource() == addBtn) result = num1 + num2;

else if (e.getSource() == subBtn) result = num1 - num2;

else if (e.getSource() == mulBtn) result = num1 \* num2;

else if (e.getSource() == divBtn) {

if (num2 != 0) result = num1 / num2;

else {

resultLabel.setText("Error: Divide by zero");

return;

}

}

resultLabel.setText("Result: " + result);

} catch (Exception ex) {

resultLabel.setText("Error: Invalid Input");

}

}

public static void main(String[] args) {

new CalculatorSwing();

}

}

**Assignment 10:**

1. Write Java program to demonstrate any five different Queue operations

import java.util.\*;

public class QueueOperations {

public static void main(String[] args) {

Queue<Integer> queue = new LinkedList<>();

queue.add(10);

queue.add(20);

queue.add(30);

System.out.println("Queue after add(): " + queue);

queue.add(40);

System.out.println("Queue after offer(): " + queue);

System.out.println("Head element (peek): " + queue.peek());

System.out.println("Removed element (poll): " + queue.poll());

System.out.println("Queue after poll(): " + queue);

System.out.println("Is queue empty? " + queue.isEmpty());

}

}

Write a java program to find factorial of a number using AWT /Swing.

import javax.swing.\*;

import java.awt.event.\*;

public class FactorialCalculator extends JFrame implements ActionListener {

JLabel label;

JTextField textField;

JButton button;

JLabel resultLabel;

public FactorialCalculator() {

setTitle("Factorial Calculator");

setSize(300, 200);

setLayout(null);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

label = new JLabel("Enter a number:");

label.setBounds(30, 30, 100, 30);

add(label);

textField = new JTextField();

textField.setBounds(140, 30, 100, 30);

add(textField);

button = new JButton("Calculate");

button.setBounds(80, 70, 120, 30);

button.addActionListener(this);

add(button);

resultLabel = new JLabel("Factorial: ");

resultLabel.setBounds(30, 110, 200, 30);

add(resultLabel);

setVisible(true);

}

public void actionPerformed(ActionEvent e) {

try {

int number = Integer.parseInt(textField.getText());

if (number < 0) {

resultLabel.setText("Factorial not defined for negatives.");

return;

}

long fact = 1;

for (int i = 1; i <= number; i++) {

fact \*= i;

}

resultLabel.setText("Factorial: " + fact);

} catch (NumberFormatException ex) {

resultLabel.setText("Please enter a valid integer.");

}

}

public static void main(String[] args) {

new FactorialCalculator();

}

}