**1.Create an Applet to accept amount in US dollars and convert it to Indian rupees.(Hint: 1 USD=82.30 Indian Rupee).**

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

/\*

<applet code="MyProgram" width=600 height=300>

</applet>

\*/

class MyProgram extends Applet implements ActionListener

{

Label l1;

Label l2;

TextField txtUS;

TextField txtINR;

Button b1;

public void init(){

l1=new Label("US currency");

txtUS=new TextField(20);

l2=new Label("INR currency");

txtINR=new TextField(20);

b1=new Button("Convert");

add(l1);

add(txtUS);

add(l2);

add(txtINR);

add(b1);

b1.addActionListener(this);

}

public void actionPerformed(ActionEvent e){

if(e.getSource()==b1){

float f1,f2;

f1=Float.parseFloat(txtUS.getText());

f2=f1\*82.30f;

txtINR.setText(Float.toString(f2));

}

}

}

**2. Write a Java program to count the number of characters in a file.**

import java.io.\*;

class fileinput

{

public static void main (String[] args) throws IOException

{

FileInputStream fis= new FileInputStream("file1.txt");

int ch,count=0;

try{

while((ch=fis.read())!= -1){

//if(ch!=' ') //if you don't want to consider spaces

count++;

}

System.out.println((count));

}

catch(Exception e){}

}

}

Create another text file file.txt

**3. Create a Window using swing component where you have a textbox to accept a number and on clicking on a button calculate its factorial and display the factorial in another textbox**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class FactorialCalculator extends JFrame {

private JTextField numberField;

private JButton calculateButton;

private JLabel resultLabel;

public FactorialCalculator() {

setTitle("Factorial Calculator");

setSize(300, 150);

setLayout(new GridLayout(3, 1));

numberField = new JTextField();

numberField.setHorizontalAlignment(JTextField.CENTER);

calculateButton = new JButton("Calculate Factorial");

calculateButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

calculateFactorial();

}

});

resultLabel = new JLabel("", JLabel.CENTER);

add(new JLabel("Enter a number:"));

add(numberField);

add(calculateButton);

add(resultLabel);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

private void calculateFactorial() {

try {

int num = Integer.parseInt(numberField.getText());

if (num < 0) {

resultLabel.setText("Please enter a non-negative integer.");

return;

}

long factorial = 1;

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

factorial \*= i;

}

resultLabel.setText("Factorial of " + num + " is: " + factorial);

} catch (NumberFormatException ex) {

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

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(new Runnable() {

public void run() {

FactorialCalculator frame = new FactorialCalculator();

frame.setVisible(true);

}

});

}

}

**4. Write a Java program to copy digits to digit.txt and characters to characters.txt and special characters to spchr.txt from a source file.**

import java.io.\*;

public class FileContentSplitter {

public static void main(String[] args) {

String sourceFileName = "source.txt";

String digitFileName = "digit.txt";

String characterFileName = "characters.txt";

String specialCharacterFileName = "spchr.txt";

try {

FileReader fileReader = new FileReader(sourceFileName);

BufferedReader bufferedReader = new BufferedReader(fileReader);

FileWriter digitFileWriter = new FileWriter(digitFileName);

FileWriter characterFileWriter = new FileWriter(characterFileName);

FileWriter spchrFileWriter = new FileWriter(specialCharacterFileName);

String line;

while ((line = bufferedReader.readLine()) != null) {

for (char c : line.toCharArray()) {

if (Character.isDigit(c)) {

digitFileWriter.write(c);

} else if (Character.isLetter(c)) {

characterFileWriter.write(c);

} else {

spchrFileWriter.write(c);

}

}

}

bufferedReader.close();

digitFileWriter.close();

characterFileWriter.close();

spchrFileWriter.close();

System.out.println("Content copied successfully!");

} catch (FileNotFoundException e) {

System.err.println("Source file not found: " + sourceFileName);

} catch (IOException e) {

System.err.println("Error reading or writing files: " + e.getMessage());

}

}

}

Same like count the character program

**5. Write a swing Frame to display the first 10 fibonacii numbers.**

import javax.swing.\*;

public class FibonacciNumbersFrame extends JFrame {

public FibonacciNumbersFrame() {

setTitle("Fibonacci Numbers");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(300, 200);

setLocationRelativeTo(null);

JTextArea textArea = new JTextArea();

textArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(textArea);

StringBuilder fibNumbers = new StringBuilder();

int prev = 0;

int curr = 1;

for (int i = 0; i < 10; i++) {

fibNumbers.append(curr).append("\n");

int next = prev + curr;

prev = curr;

curr = next;

}

textArea.setText(fibNumbers.toString());

add(scrollPane);

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> {

FibonacciNumbersFrame frame = new FibonacciNumbersFrame();

frame.setVisible(true);

});

}

}

**6. Design a Frame to convert a temperature from Celcius to Fahrenheit. (Temperature in Fahrenheit (°F) = (Temperature in degrees Celsius (°C) \* 9/5) + 32.)**

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

class tempConv extends Frame implements ActionListener

{

Label lbDeg,lbFar;

TextField tfDeg,tfFar;

Button btComp,btClear;

tempConv()

{

setLayout(new GridLayout(4,2));

setSize(400,200);

setVisible(true);

lbDeg = new Label("Enter temperature in degree

Celcius:");

add(lbDeg);

tfDeg = new TextField();

add(tfDeg);

lbFar = new Label("Temperature in Farhenheit is:");

add(lbFar);

tfFar = new TextField();

add(tfFar);

btComp = new Button("Convert Celcius to Farhenheit");

btComp.addActionListener(this);

add(btComp);

addWindowListener(new WindowAdapter()

{

public void windowClosing(WindowEvent we)

{

System.exit(0);

}

});

}

public void actionPerformed(ActionEvent ae)

{

if(ae.getSource() == btComp)

{

int celcius = Integer.parseInt(tfDeg.getText());

int farhenheit = (celcius \*9/5)+32;

tfFar.setText(String.valueOf(farhenheit));

}

if(ae.getSource() == btClear)

{

tfDeg.setText("");

tfFar.setText("");

tfDeg.requestFocus();

}

}

class apoorva

{

public static void main(String[] args)

{

new tempConv();

}

}

}

**7. Create an abstract class Emp with abstract methods get() and display(). Implement the abstract methods in a class EmpImpl which has data elements empno, ename, basic and commission. get() and display() methods should accept and display emp details. EmpImpl class should also have method calculate() which will calculate commission as 2% of basic. Instantiate EmpImpl class and call its methods.**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

// Abstract class Emp

abstract class Emp {

// Abstract method to get employee details

public abstract void get() throws IOException;

// Abstract method to display employee details

public abstract void display();

}

// Concrete subclass EmpImpl implementing Emp

class EmpImpl extends Emp {

private int empno;

private String ename;

private double basic;

private double commission;

// Implementation of the get() method

@Override

public void get() throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

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

empno = Integer.parseInt(br.readLine());

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

ename = br.readLine();

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

basic = Double.parseDouble(br.readLine());

}

// Implementation of the display() method

@Override

public void display() {

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

System.out.println("Employee Number: " + empno);

System.out.println("Employee Name: " + ename);

System.out.println("Basic Salary: " + basic);

System.out.println("Commission: " + commission);

}

// Method to calculate commission as 2% of basic

public void calculate() {

commission = 0.02 \* basic;

}

}

public class Main {

public static void main(String[] args) {

EmpImpl emp = new EmpImpl();

try {

emp.get(); // Accept employee details

emp.calculate(); // Calculate commission

emp.display(); // Display employee details

} catch (IOException e) {

System.err.println("Error reading input: " + e.getMessage());

}

}

}

**9. Accept a string in a awt frame which will accept a string and reverse it and display it.**

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class StringReversalFrame extends Frame implements ActionListener

{

private TextField inputTextField;

private Label reversedLabel;

public StringReversalFrame()

{

setTitle("String Reversal");

setSize(300, 150);

setLayout(new GridLayout(3, 1));

inputTextField = new TextField();

reversedLabel = new Label();

Button reverseButton = new Button("Reverse");

reverseButton.addActionListener(this);

add(new Label("Enter a string:"));

add(inputTextField);

add(reverseButton);

add(new Label("Reversed String:"));

add(reversedLabel);

addWindowListener(new java.awt.event.WindowAdapter() {

public void windowClosing(java.awt.event.WindowEvent windowEvent) {

System.exit(0);

}

});

}

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equals("Reverse")) {

String inputString = inputTextField.getText();

String reversedString = reverseString(inputString);

reversedLabel.setText(reversedString);

}

}

private String reverseString(String str) {

StringBuilder reversed = new StringBuilder(str);

return reversed.reverse().toString();

}

public static void main(String[] args) {

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

StringReversalFrame frame = new StringReversalFrame();

frame.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

}

**10. Accept three numbers in three text boxes and display the largest among the three on clicking on a button**.

import java.awt.event.\*;

import java.awt.\*;

class Largest\_no\_Finder extends Frame implements ActionListener {

TextField t1,t2,t3;

Label l1,l2,l3,l4;

Button cal;

Largest\_no\_Finder(){

l1=new Label("Enter the first number:");

l1.setBounds(100,80,150,30);

add(l1);

l2=new Label("Enter the second number:");

l2.setBounds(100,115,150,30);

add(l2);

l3=new Label("Enter the third number:");

l3.setBounds(100,150,150,30);

add(l3);

t1=new TextField();

t1.setBounds(255,80,150,25);

add(t1);

t2=new TextField();

t2.setBounds(255,115,150,25);

add(t2);

t3=new TextField();

t3.setBounds(255,150,150,25);

add(t3);

cal=new Button("calculate");

cal.setBounds(200,185,120,30);

add(cal);

l4=new Label("Oops! Enter the numbers");

l4.setBounds(180,220,150,30);

cal.addActionListener(this);

setLayout(null);

setSize(600,600);

setVisible(true);

this.addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

System.exit(0);

}

});

}

public void actionPerformed(ActionEvent e){

if(t1.getText().equals("") || t2.getText().equals("") || t3.getText().equals("")) {

l4.setForeground(Color.RED);

add(l4);

}

else{

int n1,n2,n3,max;

n1=Integer.parseInt(t1.getText());

n2=Integer.parseInt(t2.getText());

n3=Integer.parseInt(t3.getText());

if(n1>n2 && n1>n3)

max=n1;

else if(n2>n3)

max=n2;

else

max=n3;

l4.setText("Largest no. is "+max);

l4.setForeground(Color.GREEN);

add(l4);

}

}

}

public class p1 {

public static void main(String a[]){

new Largest\_no\_Finder();

}

}

**11.** **Create a class Arith with static methods to add, subtract, multiple and divide two numbers and returns the result. Use it in an executable class**

class Arith {

public static int add(int num1, int num2)

{

return num1 + num2;

}

public static int subtract(int num1, int num2)

{

return num1 - num2;

}

public static int multiply(int num1, int num2)

{

return num1 \* num2;

}

public static double divide(double num1, double num2)

{

if (num2 != 0)

{

return num1 / num2;

}

else

{

throw new ArithmeticException("Division by zero is not allowed.");

}

}

}

public class ArithTest

{

public static void main(String[] args)

{

int num1 = 10;

int num2 = 5;

// Call the static methods of Arith class

int sum = Arith.add(num1, num2);

int difference = Arith.subtract(num1, num2);

int product = Arith.multiply(num1, num2);

double quotient = Arith.divide(num1, num2);

System.out.println("Number 1: " + num1);

System.out.println("Number 2: " + num2);

System.out.println("Sum: " + sum);

System.out.println("Difference: " + difference);

System.out.println("Product: " + product);

System.out.println("Quotient: " + quotient);

}

}

**12. Write a program to create a class Student2 along with two method getData(),printData() to get the value through argument and display the data in printData. Create the two objects s1 ,s2 to declare and access the values from class STtest**

import java.io.\*;

class student2

{

String name;

String city;

int age;

void getdata(String sname,String scity ,int sage) throws IOException

{

name=sname;

city=scity;

age=sage;

}

void printdata()

{

System.out.println(" student name:"+name+"\n city:"+city+"\nage:"+age);

}

}

class St2

{

public static void main(String args[])throws IOException

{

String name,city;

int age;

BufferedReader br = new BufferedReader(new

InputStreamReader(System.in));

System.out.println("Enter name");

name=br.readLine();

System.out.println("Enter city");

city=br.readLine();

System.out.println("Enter age");

age=Integer.parseInt(br.readLine());

student2 s1=new student2();

s1.getdata(name,city,age);

s1.printdata();

System.out.println("Enter name");

name=br.readLine();

System.out.println("Enter city");

city=br.readLine();

System.out.println("Enter age");

age=Integer.parseInt(br.readLine());

student2 s2=new student2();

s2.getdata(name,city,age);

s2.printdata();

}

}

**13.** **Overload area() method to calculate the area of three different shapes**

class ShapeAreaCalculator

{

// Area of a rectangle

public double area(double length, double width)

{

return length \* width;

}

// Area of a circle

public double area(double radius)

{

return Math.PI \* radius \* radius;

}

// Area of a triangle

public double area(double base, double height)

{

return 0.5 \* base \* height;

}

}

public class AreaCalculatorTest

{

public static void main(String[] args)

{

ShapeAreaCalculator calculator = new ShapeAreaCalculator();

// Rectangle

double length = 5.0;

double width = 3.0;

double rectangleArea = calculator.area(length, width);

System.out.println("Area of Rectangle: " + rectangleArea);

// Circle

double radius = 4.0;

double circleArea = calculator.area(radius);

System.out.println("Area of Circle: " + circleArea);

// Triangle

double base = 6.0;

double height = 2.5;

double triangleArea = calculator.area(base, height);

System.out.println("Area of Triangle: " + triangleArea);

}

}

**14. Write a Java program to calculate total, average and grade for 10 students. Consider the necessary datamembers and member methods required to achieve the requirements**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

class Student {

private String name;

private int[] marks;

public Student(String name)

{

this.name = name;

marks = new int[5]; // Assuming 5 subjects

}

public void setMarks(int[] marks)

{

this.marks = marks;

}

public int getTotalMarks()

{

int total = 0;

for (int mark : marks)

{

total += mark;

}

return total;

}

public double getAverageMarks()

{

int total = getTotalMarks();

return (double) total / marks.length;

}

public String getGrade()

{

double average = getAverageMarks();

if (average >= 90)

{

return "A";

}

else if (average >= 80)

{

return "B";

}

else if (average >= 70)

{

return "C";

}

else if (average >= 60)

{

return "D";

}

else

{

return "F";

}

}

public String getName()

{

return name;

}

}

public class StudentGradeCalculator

{

public static void main(String[] args) throws IOException

{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

Student[] students = new Student[10];

// Input marks for each student

for (int i = 0; i < 10; i++)

{

System.out.print("Enter name of student " + (i + 1) + ": ");

String name = br.readLine();

students[i] = new Student(name);

System.out.println("Enter marks for student " + (i + 1) + ": ");

int[] marks = new int[5]; // Assuming 5 subjects

for (int j = 0; j < 5; j++)

{

System.out.print("Enter marks for subject " + (j + 1) + ": ");

marks[j] = Integer.parseInt(br.readLine());

}

students[i].setMarks(marks);

}

// Display results for each student

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

for (Student student : students) {

System.out.println("Name: " + student.getName());

System.out.println("Total Marks: " + student.getTotalMarks());

System.out.println("Average Marks: " + student.getAverageMarks());

System.out.println("Grade: " + student.getGrade());

System.out.println();

}

}

}

**15. Write a program to create a class named shape. In this class we have three sub classes circle, triangle and square each class has two member function named draw () and erase (). Create these using polymorphism concepts.**

// Shape class (parent class)

class Shape

{

// Common methods for all subclasses

public void draw()

{

System.out.println("Drawing a shape");

}

public void erase()

{

System.out.println("Erasing a shape");

}

}

// Circle class (subclass of Shape)

class Circle extends Shape

{

@Override

public void draw()

{

System.out.println("Drawing a circle");

}

@Override

public void erase()

{

System.out.println("Erasing a circle");

}

}

// Triangle class (subclass of Shape)

class Triangle extends Shape {

@Override

public void draw()

{

System.out.println("Drawing a triangle");

}

@Override

public void erase()

{

System.out.println("Erasing a triangle");

}

}

// Square class (subclass of Shape)

class Square extends Shape

{

@Override

public void draw()

{

System.out.println("Drawing a square");

}

@Override

public void erase()

{

System.out.println("Erasing a square");

}

}

public class Main

{

public static void main(String[] args)

{

// Using polymorphism to call draw and erase methods

Shape shape1 = new Circle();

Shape shape2 = new Triangle();

Shape shape3 = new Square();

shape1.draw();

shape1.erase();

shape2.draw();

shape2.erase();

shape3.draw();

shape3.erase();

}

}

**16. Create Emp class in a package payroll with data elements empno and ename and methods get() and display() which will accept and display empno and ename. Inherit Emp class in Clerk class with data element typingspeed and override get() and display() to accept and display empno, ename and typingspeed. Create an instance of Clerk class in an executable class and call its methods.**

**17.** **Create user defined exception that will be thrown when basic is not between 2000 and 7000.**

**18. Write a program to demonstrate synchronization**

public class SynchronizationDemo

{

public static void main(String[] args)

{

SharedCounter sharedCounter = new SharedCounter();

// Create three threads and start them

Thread thread1 = new CounterThread(sharedCounter);

Thread thread2 = new CounterThread(sharedCounter);

Thread thread3 = new CounterThread(sharedCounter);

thread1.start();

thread2.start();

thread3.start();

// Wait for all threads to complete their execution

try

{

thread1.join();

thread2.join();

thread3.join();

}

catch (InterruptedException e)

{

e.printStackTrace();

}

// Print the final value of the counter

System.out.println("Final counter value: " + sharedCounter.getCounter());

}

}

// SharedCounter class to hold the shared counter

class SharedCounter

{

private int counter;

public SharedCounter()

{

this.counter = 0;

}

// Method to increment the counter (without synchronization)

public void increment()

{

int temp = counter;

temp++;

counter = temp;

}

// Method to get the counter value

public int getCounter()

{

return counter;

}

}

// CounterThread class to represent a thread that increments the shared counter

class CounterThread extends Thread

{

private SharedCounter sharedCounter;

public CounterThread(SharedCounter sharedCounter)

{

this.sharedCounter = sharedCounter;

}

public void run() {

for (int i = 0; i < 100000; i++)

{

sharedCounter.increment();

}

}

}