Experiment 1 (a) code

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
import java.util.Scanner;

class Factorial {
    int n;

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

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Factorial obj = new Factorial();
        System.out.print("Enter a number: ");
        obj.n = sc.nextInt();
        int result = obj.computeFactorial(obj.n);
        System.out.println("The factorial of " + obj.n + " is: " + result);
    }
}
```

Output :

```
Enter a number: 5
The factorial of 5 is: 120
```

Experiment 1 (b) code

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
import java.util.Scanner;

class PrimeNumbers {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of prime numbers to display: ");
        int n = sc.nextInt();

        int count = 0, num = 2;

        while (count < n) {
            boolean isPrime = true;
            for (int i = 2; i <= Math.sqrt(num); i++) {
                if (num % i == 0) {
                    isPrime = false;
                    break;
                }
            }

            if (isPrime) {
                System.out.print(num + " ");
                count++;
            }
            num++;
        }
    }
}
```

Output:

```
Enter the number of prime numbers to display: 5
2 3 5 7 11
```

Experiment 1 (c) code

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
import java.util.Scanner;

class SumAvg {

    int computeSum(int n) {
        int sum = 0;
        for (int i = 1; i <= n; i++) {
            sum += i;
        }
        return sum;
    }

    double computeAvg(int sum, int n) {
        return (double) sum / n;
    }

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the value of n: ");
        int n = sc.nextInt();

        SumAvg obj = new SumAvg();
        int sum = obj.computeSum(n);
        double avg = obj.computeAvg(sum, n);

        System.out.println("Sum: " + sum);
        System.out.println("Average: " + avg);
    }
}
```

Output:

```
Enter the value of n: 5
Sum: 15
Average: 3.0
```

Experiment 2

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
import java.util.Scanner;

class Calculator {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        boolean continueCalc = true;

        while (continueCalc) {
            System.out.print("Enter first number: ");
            int num1 = sc.nextInt();

            System.out.print("Enter second number: ");
            int num2 = sc.nextInt();

            System.out.println("Choose an operation:");
            System.out.println("1. Add");
            System.out.println("2. Subtract");
            System.out.println("3. Multiply");
            System.out.println("4. Divide");
            System.out.println("5. Factorial (of first number)");
            int choice = sc.nextInt();

            switch (choice) {
                case 1:
                    System.out.println("Sum: " + add(num1, num2));
                    break;
                case 2:
                    System.out.println("Difference: " + subtract(num1, num2));
                    break;
                case 3:
                    System.out.println("Product: " + multiply(num1, num2));
                    break;
                case 4:
                    if (num2 != 0) {
                        System.out.println("Quotient: " + divide(num1, num2));
                    } else {
                        System.out.println("Error: Division by zero is not allowed.");
```

```
                }
                break;
            case 5:
                System.out.println("Factorial: " + factorial(num1));
                break;
            default:
                System.out.println("Invalid choice!");
                break;
        }

        System.out.print("Do you want to perform another calculation? (yes/no): ");
        String response = sc.next();
        continueCalc = response.equalsIgnoreCase("yes");
    }
  }

  static int add(int a, int b) {
    return a + b;
  }

  static int subtract(int a, int b) {
    return a - b;
  }

  static int multiply(int a, int b) {
    return a * b;
  }

  static double divide(int a, int b) {
    return (double) a / b;
  }

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

Output:

```
Enter first number: 5
Enter second number: 3
Choose an operation:
1. Add
2. Subtract
3. Multiply
4. Divide
5. Factorial (of first number)
3
Product: 15
Do you want to perform another calculation? (yes/no): yes

Enter first number: 4
Enter second number: 0
Choose an operation:
1. Add
2. Subtract
3. Multiply
4. Divide
5. Factorial (of first number)
4
Error: Division by zero is not allowed.
Do you want to perform another calculation? (yes/no): no
```

Experiment 3

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
class Rectangle {
    int width, length;
    String color;
    int area;

    Rectangle(int width, int length, String color) {
        this.width = width;
        this.length = length;
        this.color = color;
    }

    int rectArea() {
        area = width * length;
        return area;
    }
}

public class Main {
    public static void main(String[] args) {
        Rectangle rect1 = new Rectangle(4, 5, "Red");
        Rectangle rect2 = new Rectangle(3, 6, "Red");

        int area1 = rect1.rectArea();
        int area2 = rect2.rectArea();

        if (area1 == area2 && rect1.color.equals(rect2.color)) {
            System.out.println("Matching Rectangles");
        } else {
            System.out.println("Non-Matching Rectangles");
        }
    }
}
```

Output:

```
Non-Matching Rectangles
```

Experiment 4

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
class Adder {
    int num1, num2, num3;

    Adder(int num1, int num2) {
        this.num1 = num1;
        this.num2 = num2;
    }

    Adder(int num1, int num2, int num3) {
        this.num1 = num1;
        this.num2 = num2;
        this.num3 = num3;
    }

    int add(int a, int b) {
        return a + b;
    }

    int add(int a, int b, int c) {
        return a + b + c;
    }
}
public class TestAdder {
    public static void main(String[] args) {
        Adder adder1 = new Adder(5, 10);
        int sum1 = adder1.add(adder1.num1, adder1.num2);

        Adder adder2 = new Adder(3, 6, 9);
        int sum2 = adder2.add(adder2.num1, adder2.num2, adder2.num3);

        System.out.println("Sum of first two numbers: " + sum1);
        System.out.println("Sum of all three numbers: " + sum2);
    }
}
```

Output:

```
Sum of first two numbers: 15
Sum of all three numbers: 18
```

Experiment 5

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
import java.util.Arrays;
import java.util.Scanner;

class ArraySort {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter size of integer array: ");
        int intSize = sc.nextInt();
        int[] intArray = new int[intSize];
        System.out.println("Enter elements of integer array:");
        for (int i = 0; i < intSize; i++) {
            intArray[i] = sc.nextInt();
        }

        System.out.print("Enter size of string array: ");
        int strSize = sc.nextInt();
        String[] strArray = new String[strSize];
        System.out.println("Enter elements of string array:");
        for (int i = 0; i < strSize; i++) {
            strArray[i] = sc.next();
        }

        Arrays.sort(intArray);
        Arrays.sort(strArray);

        System.out.println("Sorted Integer Array: " + Arrays.toString(intArray));
        System.out.println("Sorted String Array: " + Arrays.toString(strArray));
    }
}
```

Output:

```
Enter size of integer array: 4
Enter elements of integer array:
7
3
9
1
Sorted Integer Array: [1, 3, 7, 9]

Enter size of string array: 3
Enter elements of string array:
apple
banana
cherry
Sorted String Array: [apple, banana, cherry]
```

Experiment 6

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
import java.util.Scanner;

class MatrixAddition {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of rows for first matrix: ");
        int rows1 = sc.nextInt();
        System.out.print("Enter number of columns for first matrix: ");
        int cols1 = sc.nextInt();
        int[][] matrix1 = new int[rows1][cols1];

        System.out.println("Enter elements of the first matrix:");
        for (int i = 0; i < rows1; i++) {
            for (int j = 0; j < cols1; j++) {
                matrix1[i][j] = sc.nextInt();
            }
        }

        System.out.print("Enter number of rows for second matrix: ");
        int rows2 = sc.nextInt();
        System.out.print("Enter number of columns for second matrix: ");
        int cols2 = sc.nextInt();
        int[][] matrix2 = new int[rows2][cols2];

        System.out.println("Enter elements of the second matrix:");
        for (int i = 0; i < rows2; i++) {
            for (int j = 0; j < cols2; j++) {
                matrix2[i][j] = sc.nextInt();
            }
        }

        if (rows1 != rows2 || cols1 != cols2) {
            System.out.println("Addition of given matrices not possible due to size mismatch");
        } else {
            int[][] sumMatrix = new int[rows1][cols1];
            for (int i = 0; i < rows1; i++) {
                for (int j = 0; j < cols1; j++) {
                    sumMatrix[i][j] = matrix1[i][j] + matrix2[i][j];
```

```
            }
        }

        System.out.println("First Matrix:");
        displayMatrix(matrix1);
        System.out.println("Second Matrix:");
        displayMatrix(matrix2);
        System.out.println("Sum Matrix:");
        displayMatrix(sumMatrix);
    }
  }

  static void displayMatrix(int[][] matrix) {
    for (int[] row : matrix) {
        for (int element : row) {
            System.out.print(element + " ");
        }
        System.out.println();
    }
  }
}
```

Output:

```
Enter number of rows for first matrix: 2
Enter number of columns for first matrix: 2
Enter elements of the first matrix:
1
2
3
4
Enter number of rows for second matrix: 2
Enter number of columns for second matrix: 2
Enter elements of the second matrix:
5
6
7
8
First Matrix:
1 2
3 4
Second Matrix:
5 6
7 8
Sum Matrix:
6 8
10 12
```

Experiment 7

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
class Player {
    String name;
    int age;
    int ranking;

    Player(String name, int age, int ranking) {
        this.name = name;
        this.age = age;
        this.ranking = ranking;
    }

    void display() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Ranking: " + ranking);
    }
}

class Cricket_Player extends Player {
    String game;
    String role;

    Cricket_Player(String name, int age, int ranking, String game, String role) {
        super(name, age, ranking);
        this.game = game;
        this.role = role;
    }

    void display() {
        super.display();
        System.out.println("Game: " + game);
        System.out.println("Role: " + role);
    }
}

class Football_Player extends Player {
    String game;
    String place;
```

```
    Football_Player(String name, int age, int ranking, String game, String place) {
        super(name, age, ranking);
        this.game = game;
        this.place = place;
    }

    void display() {
        super.display();
        System.out.println("Game: " + game);
        System.out.println("Place: " + place);
    }
}

class Hockey_Player extends Player {
    String game;
    String position;

    Hockey_Player(String name, int age, int ranking, String game, String position) {
        super(name, age, ranking);
        this.game = game;
        this.position = position;
    }

    void display() {
        super.display();
        System.out.println("Game: " + game);
        System.out.println("Position: " + position);
    }
}

public class Main {
    public static void main(String[] args) {
        Cricket_Player cricketPlayer = new Cricket_Player("Sachin Tendulkar", 50, 1, "Cricket",
"Batsman");
        Football_Player footballPlayer = new Football_Player("Lionel Messi", 36, 1, "Football",
"Forward");
        Hockey_Player hockeyPlayer = new Hockey_Player("Dhyan Chand", 74, 1, "Hockey",
"Forward");

        System.out.println("Cricket Player:");
        cricketPlayer.display();
        System.out.println();

        System.out.println("Football Player:");
        footballPlayer.display();
        System.out.println();
```

```
        System.out.println("Hockey Player:");
        hockeyPlayer.display();
    }
}
```

Output:

```
Cricket Player:
Name: Sachin Tendulkar
Age: 50
Ranking: 1
Game: Cricket
Role: Batsman

Football Player:
Name: Lionel Messi
Age: 36
Ranking: 1
Game: Football
Place: Forward

Hockey Player:
Name: Dhyan Chand
Age: 74
Ranking: 1
Game: Hockey
Position: Forward
```

Experiment 8

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
interface Shape {
    double pi = 3.14159;
    double area();
    double perimeter();
}

class Circle implements Shape {
    double radius;

    Circle(double radius) {
        this.radius = radius;
    }

    public double area() {
        return pi * radius * radius;
    }

    public double perimeter() {
        return 2 * pi * radius;
    }
}
class Rectangle implements Shape {
    double length;
    double width;

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

    public double area() {
        return length * width;
    }

    public double perimeter() {
        return 2 * (length + width);
    }
}
```

```
class Ellipse implements Shape {
    double major;
    double minor;

    Ellipse(double major, double minor) {
        this.major = major;
        this.minor = minor;
    }

    public double area() {
        return pi * major * minor;
    }

    public double perimeter() {
        return 2 * pi * Math.sqrt((major * major + minor * minor) / 2);
    }
}

public class Main {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        Rectangle rectangle = new Rectangle(4, 6);
        Ellipse ellipse = new Ellipse(3, 2);

        System.out.println("Circle Area: " + circle.area());
        System.out.println("Circle Perimeter: " + circle.perimeter());
        System.out.println();

        System.out.println("Rectangle Area: " + rectangle.area());
        System.out.println("Rectangle Perimeter: " + rectangle.perimeter());
        System.out.println();

        System.out.println("Ellipse Area: " + ellipse.area());
        System.out.println("Ellipse Perimeter: " + ellipse.perimeter());
    }
}
```

Output:

```
Circle Area: 78.53975
Circle Perimeter: 31.4159

Rectangle Area: 24.0
Rectangle Perimeter: 20.0

Ellipse Area: 18.84954
Ellipse Perimeter: 10.726493
```

Experiment 9

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
class ExceptionHandling {
    public static void main(String[] args) {
        int a = 10, b = 0, c;
        int[] arr = new int[5];

        try {
            c = a / b;
            System.out.println("Value of c: " + c);
            System.out.println("Accessing arr[6]: " + arr[6]);
        } catch (ArithmeticException e) {
            System.out.println("Arithmetic Exception: Division by zero");
            e.printStackTrace();
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Array Index Out Of Bounds Exception: Accessing invalid index");
            e.printStackTrace();
        } finally {
            System.out.println("Finally Block Executed");
        }
    }
}
```

Output:

```
Arithmetic Exception: Division by zero
java.lang.ArithmeticException: / by zero
    at ExceptionHandling.main(ExceptionHandling.java:5)
Array Index Out Of Bounds Exception: Accessing invalid index
java.lang.ArrayIndexOutOfBoundsException: Index 6 out of bounds for length 5
    at ExceptionHandling.main(ExceptionHandling.java:7)
Finally Block Executed
```

Experiment 11

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
import java.io.*;

class FileHandling {
    public static void main(String[] args) {
        try {
            FileReader fileReader = new FileReader("input.txt");
            BufferedReader bufferedReader = new BufferedReader(fileReader);
            FileWriter fileWriter = new FileWriter("output.txt");
            BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);

            String line;
            while ((line = bufferedReader.readLine()) != null) {
                bufferedWriter.write(line);
                bufferedWriter.newLine();
            }

            bufferedReader.close();
            bufferedWriter.close();
            fileReader.close();
            fileWriter.close();
        } catch (IOException e) {
            System.out.println("An error occurred: " + e.getMessage());
        }
    }
}
```

Output:

```
Hello, this is the first line.
This is the second line.
And this is the third line.
```

Experiment 12

Name: Rutik Kawade
Roll No: 304A064
Batch:A4

```
class NumberThread extends Thread {
    private final Object lock;

    public NumberThread(Object lock) {
        this.lock = lock;
    }

    public void run() {
        synchronized (lock) {
            for (int i = 1; i <= 5; i++) {
                System.out.println(i);
                try {
                    Thread.sleep(500);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                lock.notify();
                try {
                    lock.wait();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
            lock.notify();
        }
    }
}

class AlphabetThread extends Thread {
    private final Object lock;

    public AlphabetThread(Object lock) {
        this.lock = lock;
    }

    public void run() {
        synchronized (lock) {
            for (char c = 'A'; c <= 'E'; c++) {
                System.out.println(c);
```

```
                try {
                    Thread.sleep(500);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                lock.notify();
                try {
                    lock.wait();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
            lock.notify();
        }
    }
}

public class MultiThreading {
    public static void main(String[] args) {
        Object lock = new Object();
        NumberThread numberThread = new NumberThread(lock);
        AlphabetThread alphabetThread = new AlphabetThread(lock);

        numberThread.start();
        alphabetThread.start();
    }
}
```

Output:

```
1
A
2
B
3
C
4
D
5
E
```