1.Create a program to find the factors of a number taken as user input, store the factors in an array, and display the factors. Also find the sum, sum of square of factors and product of the factors and display the results

Hint =>

1. Take the input for a number
2. Write a static Method to find the factors of the number and save them in an array and return the array.
3. To find factors and save to array will have two loops. The first loop to find the count and initialize the array with the count. And the second loop save the factors into the array
4. Write a method to find the sum of the factors using factors array
5. Write a method to find the product of the factors using factors array
6. Write a method to find the sum of square of the factors using Math.pow() method

```
import java.util.Scanner;

public class FactorAnalysis {

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

        System.out.print("Enter a positive integer: ");

        if (!input.hasNextInt()) {
            System.out.println("Invalid input. Please enter an integer.");
            input.close();
            return;a
        }

        int number = input.nextInt();

        if (number <= 0) {
            System.out.println("Please enter a positive integer.");
            input.close();
            return;
        }

        int[] factors = findFactors(number);

        System.out.print("Factors of " + number + " are: ");
        for (int factor : factors) {
            System.out.print(factor + " ");
        }
        System.out.println();

        int sum = sumOfFactors(factors);
        long product = productOfFactors(factors);
        double sumOfSquares = sumOfSquaresOfFactors(factors);

        System.out.println("Sum of factors: " + sum);
        System.out.println("Product of factors: " + product);
        System.out.println("Sum of squares of factors: " + sumOfSquares);
```

```
        input.close();
    }

    public static int[] findFactors(int number) {
        int count = 0;
        for (int i = 1; i <= number; i++) {
            if (number % i == 0) {
                count++;
            }
        }

        int[] factors = new int[count];
        int index = 0;
        for (int i = 1; i <= number; i++) {
            if (number % i == 0) {
                factors[index] = i;
                index++;
            }
        }
        return factors;
    }

    public static int sumOfFactors(int[] factors) {
        int sum = 0;
        for (int factor : factors) {
            sum += factor;
        }
        return sum;
    }

    public static long productOfFactors(int[] factors) {
        long product = 1;
        for (int factor : factors) {
            product *= factor;
        }
        return product;
    }

    public static double sumOfSquaresOfFactors(int[] factors) {
        double sumOfSquares = 0;
        for (int factor : factors) {
            sumOfSquares += Math.pow(factor, 2);
        }
        return sumOfSquares;
    }
}
```

2. Write a program to find the sum of n natural numbers using recursive method and compare the result with the formulae n*(n+1)/2 and show the result from both computations is correct.
Hint =>
Take the user input number and check whether it's a Natural number, if not exit

Write a Method to find the sum of n natural numbers using recursion
Write a Method to find the sum of n natural numbers using the formulae n*(n+1)/2
Compare the two results and print the result

```
import java.util.Scanner;

public class SumOfNaturalNumbers {

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

        System.out.print("Enter a positive integer (n): ");

        if (!input.hasNextInt()) {
            System.out.println("Invalid input. Please enter an integer.");
            input.close();
            return;
        }

        int n = input.nextInt();

        if (n <= 0) {
            System.out.println("Please enter a positive integer.");
            input.close();
            return;
        }

        int recursiveSum = recursiveSum(n);
        int formulaSum = formulaSum(n);

        System.out.println("Sum using recursion: " + recursiveSum);
        System.out.println("Sum using formula: " + formulaSum);

        if (recursiveSum == formulaSum) {
            System.out.println("Both results are correct and equal.");
        } else {
            System.out.println("Results do not match.");
        }

        input.close();
    }

    public static int recursiveSum(int n) {
        if (n == 1) {
            return 1;
        } else {
            return n + recursiveSum(n - 1);
        }
    }

    public static int formulaSum(int n) {
        return n * (n + 1) / 2;
```

```
    }
}
```

3. Write a program that takes a year as input and outputs the Year is a Leap Year or not

Hint =>

1. The LeapYear program only works for year >= 1582, corresponding to a year in the Gregorian calendar.
2. Also Leap year is divisible by 4 and not divisible by 100 or divisible by 400
3. Write a method to check for Leap Year using the conditions a and b

```
import java.util.Scanner;

public class LeapYearChecker {

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

        System.out.print("Enter a year (>= 1582): ");

        if (!input.hasNextInt()) {
            System.out.println("Invalid input. Please enter an integer.");
            input.close();
            return;
        }

        int year = input.nextInt();

        if (year < 1582) {
            System.out.println("Year must be 1582 or later.");
            input.close();
            return;
        }

        if (isLeapYear(year)) {
            System.out.println(year + " is a Leap Year.");
        } else {
            System.out.println(year + " is not a Leap Year.");
        }

        input.close();
    }

    public static boolean isLeapYear(int year) {
        return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
    }
}
```

4. Extend or Create a ***UnitConvertor*** utility class similar to the one shown in the notes to do the following. Please define ***static*** methods for all the UnitConvertor class methods. E.g. ***public static double convertKmToMiles(double km)*** **=>**

1. Method To convert kilometers to miles and return the value. Use the following code `double km2miles = 0.621371;`

2. Method to convert miles to kilometers and return the value. Use the following code `double miles2km = 1.60934;`
3. Method to convert meters to feet and return the value. Use the following code to convert `double meters2feet = 3.28084;`
4. Method to convert feet to meters and return the value. Use the following code to convert `double feet2meters = 0.3048;`

```
public class UnitConverter {

    public static double convertKmToMiles(double km) {
        double km2miles = 0.621371;
        return km * km2miles;
    }

    public static double convertMilesToKm(double miles) {
        double miles2km = 1.60934;
        return miles * miles2km;
    }

    public static double convertMetersToFeet(double meters) {
        double meters2feet = 3.28084;
        return meters * meters2feet;
    }

    public static double convertFeetToMeters(double feet) {
        double feet2meters = 0.3048;
        return feet * feet2meters;
    }

    public static void main(String[] args) {
        // Example usage
        double kilometers = 10;
        double miles = convertKmToMiles(kilometers);
        System.out.println(kilometers + " kilometers = " + miles + " miles");

        double milesInput = 5;
        double kilometersOutput = convertMilesToKm(milesInput);
        System.out.println(milesInput + " miles = " + kilometersOutput + " kilometers");

        double meters = 20;
        double feet = convertMetersToFeet(meters);
        System.out.println(meters + " meters = " + feet + " feet");

        double feetInput = 30;
        double metersOutput = convertFeetToMeters(feetInput);
        System.out.println(feetInput + " feet = " + metersOutput + " meters");
    }
}
```

5.Extend or Create a ***UnitConvertor*** utility class similar to the one shown in the notes to do the following. Please define ***static*** methods for all the UnitConvertor class methods. E.g.

***public static double convertYardsToFeet(double yards)* =>**

1. Method to convert yards to feet and return the value. Use following code to convert `double yards2feet = 3;`
2. Method to convert feet to yards and return the value. Use following code to convert `double feet2yards = 0.333333;`
3. Method to convert meters to inches and return the value. Use following code to convert `double meters2inches = 39.3701;`
4. Method to convert inches to meters and return the value. Use following code to convert `double inches2meters = 0.0254;`

Method to convert inches to centimeters and return the value. Use the following code `double inches2cm = 2.54;`

```
public class MeasurementConverter {

    public static double convertYardsToFeet(double yards) {
        double yards2feet = 3;
        return yards * yards2feet;
    }

    public static double convertFeetToYards(double feet) {
        double feet2yards = 0.333333;
        return feet * feet2yards;
    }

    public static double convertMetersToInches(double meters) {
        double meters2inches = 39.3701;
        return meters * meters2inches;
    }

    public static double convertInchesToMeters(double inches) {
        double inches2meters = 0.0254;
        return inches * inches2meters;
    }
```

```
    public static double convertInchesToCentimeters(double inches) {
        double inches2cm = 2.54;
        return inches * inches2cm;
    }

    public static void main(String[] args) {
        // Example usage
        double yards = 10;
        double feet = convertYardsToFeet(yards);
        System.out.println(yards + " yards = " + feet + " feet");

        double feetInput = 30;
        double yardsOutput = convertFeetToYards(feetInput);
        System.out.println(feetInput + " feet = " + yardsOutput + " yards");

        double meters = 5;
        double inches = convertMetersToInches(meters);
        System.out.println(meters + " meters = " + inches + " inches");

        double inchesInput = 12;
        double metersOutput = convertInchesToMeters(inchesInput);
        System.out.println(inchesInput + " inches = " + metersOutput + " meters");

        double inchesCm = 10;
        double centimeter = convertInchesToCentimeters(inchesCm);
        System.out.println(inchesCm + " inches = " + centimeter + " centimeters");
    }
}
```

6.Extend or Create a ***UnitConvertor*** utility class similar to the one shown in the notes to do the following. Please define ***static*** methods for all the UnitConvertor class methods. E.g.

public static double convertFarhenheitToCelsius(double farhenheit) **=>**

1. Method to convert Fahrenheit to Celsius and return the value. Use the following code `double farhenheit2celsius = (farhenheit - 32) * 5 / 9;`
2. Method to convert Celsius to Fahrenheit and return the value. Use the following code `double celsius2farhenheit = (celsius * 9 / 5) + 32;`
3. Method to convert pounds to kilograms and return the value. Use the following code `double pounds2kilograms = 0.453592;`
4. Method to convert kilograms to pounds and return the value. Use the following code `double kilograms2pounds = 2.20462;`
5. Method to convert gallons to liters and return the value. Use following code to convert `double gallons2liters = 3.78541;`

Method to convert liters to gallons and return the value. Use following code to convert `double liters2gallons = 0.264172;`

```
public class MeasurementConverter2 {

  public static double convertFahrenheitToCelsius(double fahrenheit) {
    return (fahrenheit - 32) * 5.0 / 9.0;
  }

  public static double convertCelsiusToFahrenheit(double celsius) {
    return (celsius * 9.0 / 5.0) + 32;
  }

  public static double convertPoundsToKilograms(double pounds) {
    return pounds * 0.453592;
  }

  public static double convertKilogramsToPounds(double kilograms) {
    return kilograms * 2.20462;
  }

  public static double convertGallonsToLiters(double gallons) {
    return gallons * 3.78541;
  }

  public static double convertLitersToGallons(double liters) {
    return liters * 0.264172;
```

```
    }

    public static void main(String[] args) {
        // Example usage
        double fahrenheit = 68;
        double celsius = convertFahrenheitToCelsius(fahrenheit);
        System.out.println(fahrenheit + " Fahrenheit = " + celsius + " Celsius");

        double celsiusInput = 20;
        double fahrenheitOutput = convertCelsiusToFahrenheit(celsiusInput);
        System.out.println(celsiusInput + " Celsius = " + fahrenheitOutput + " Fahrenheit");

        double pounds = 150;
        double kilograms = convertPoundsToKilograms(pounds);
        System.out.println(pounds + " pounds = " + kilograms + " kilograms");

        double kilogramsInput = 70;
        double poundsOutput = convertKilogramsToPounds(kilogramsInput);
        System.out.println(kilogramsInput + " kilograms = " + poundsOutput + " pounds");

        double gallons = 10;
        double liters = convertGallonsToLiters(gallons);
        System.out.println(gallons + " gallons = " + liters + " liters");

        double litersInput = 40;
        double gallonsOutput = convertLitersToGallons(litersInput);
        System.out.println(litersInput + " liters = " + gallonsOutput + " gallons");
    }
}
```

7.Write a program to take user input for the age of all 10 students in a class and check whether the student can vote depending on his/her age is greater or equal to 18.

Hint =>

1. Create a class ***public class StudentVoteChecker*** and define a method ***public boolean canStudentVote(int age)*** which takes in age as a parameter and returns true or false

2. Inside the method firstly validate the age for a negative number, if a negative return is false cannot vote. For valid age check for age is 18 or above return true; else return false;
3. In the main function define an array of 10 integer elements, loop through the array by take user input for the student's age, call canStudentVote() and display the result

```
import java.util.Scanner;

public class StudentVoteEligibility {

    public static boolean canStudentVote(int age) {
        if (age < 0) {
            return false; // Cannot vote if age is negative
        }
        return age >= 18; // Can vote if age is 18 or above
    }

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int[] studentAges = new int[10];

        System.out.println("Enter the ages of 10 students:");

        for (int i = 0; i < 10; i++) {
            boolean validInput = false;
            while (!validInput) {
                System.out.print("Enter age of student " + (i + 1) + ": ");
                if (input.hasNextInt()) {
                    studentAges[i] = input.nextInt();
                    validInput = true;
                } else {
                    System.out.println("Invalid input. Please enter an integer.");
                    input.next(); // Clear invalid input
                }
```

```
        }
    }

    System.out.println("\nVoting Eligibility:");
    for (int i = 0; i < 10; i++) {
        System.out.println("Student " + (i + 1) + ": " + (canStudentVote(studentAges[i]) ? "Can vote" : "Cannot vote"));
    }

    input.close();
  }
}
```

8.Create a program to find the youngest friends among 3 Amar, Akbar and Anthony based on their ages and tallest among the friends based on their heights and display it

Hint =>

1. Take user input for age and height for the 3 friends and store it in two arrays each to store the values for age and height of the 3 friends
2. Write a Method to find the youngest of the 3 friends
3. Write a Method to find the tallest of the 3 friends

```
import java.util.Scanner;

public class FriendComparison {

  public static int findYoungest(int[] ages) {
    int youngestIndex = 0;
    for (int i = 1; i < ages.length; i++) {
      if (ages[i] < ages[youngestIndex]) {
        youngestIndex = i;
      }
    }
    return youngestIndex;
  }
```

```
public static int findTallest(double[] heights) {
    int tallestIndex = 0;
    for (int i = 1; i < heights.length; i++) {
        if (heights[i] > heights[tallestIndex]) {
            tallestIndex = i;
        }
    }
    return tallestIndex;
}

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

    String[] names = {"Amar", "Akbar", "Anthony"};
    int[] ages = new int[3];
    double[] heights = new double[3];

    // Input ages and heights
    for (int i = 0; i < 3; i++) {
        boolean validInput = false;
        while(!validInput) {
            System.out.print("Enter age for " + names[i] + ": ");
            if (input.hasNextInt()) {
                ages[i] = input.nextInt();
                validInput = true;
            } else {
                System.out.println("Invalid input. Enter an integer");
                input.next();
            }
        }
```

```
            validInput = false;
            while(!validInput) {
                System.out.print("Enter height (in meters) for " + names[i] + ": ");
                if (input.hasNextDouble()) {
                    heights[i] = input.nextDouble();
                    validInput = true;
                } else {
                    System.out.println("Invalid input. Enter a double");
                    input.next();
                }
            }
        }

        int youngestIndex = findYoungest(ages);
        int tallestIndex = findTallest(heights);

        System.out.println("\nYoungest friend: " + names[youngestIndex] + " (Age: " +
ages[youngestIndex] + ")");
        System.out.println("Tallest friend: " + names[tallestIndex] + " (Height: " +
heights[tallestIndex] + " meters)");

        input.close();
    }
}
```

9.Write a program to take user input for 5 numbers and check whether a number is positive or negative. Further for positive numbers check if the number is even or odd. Finally compare the first and last elements of the array and display if they are equal, greater, or less

Hint =>

1. Write a Method to Check whether the number is positive or negative
2. Write a Method to check whether the number is even or odd
3. Write a Method to compare two numbers and return 1 if number1 > number2 or 0 if both are equal or -1 if number1 < number2
4. In the main program, Loop through the array using the length call the method ***isPositive()*** and if positive call method ***isEven()*** and print accordingly
5. If the number is negative, print negative.

Finally compare the first and last element of the array by calling the method ***compare()*** and display if they are equal, greater, or less

```
import java.util.Scanner;

public class NumberAnalysis {

    public static boolean isPositive(int number) {
        return number > 0;
    }

    public static boolean isEven(int number) {
        return number % 2 == 0;
    }

    public static int compare(int number1, int number2) {
        if (number1 > number2) {
            return 1;
        } else if (number1 == number2) {
            return 0;
        } else {
            return -1;
        }
    }

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int[] numbers = new int[5];

        System.out.println("Enter 5 numbers:");

        for (int i = 0; i < numbers.length; i++) {
            boolean validInput = false;
            while(!validInput) {
                System.out.print("Enter number " + (i + 1) + ": ");
```

```
        if (input.hasNextInt()) {
            numbers[i] = input.nextInt();
            validInput = true;
        } else {
            System.out.println("Invalid input. Enter an integer");
            input.next();
        }
    }
}

System.out.println("\nNumber Analysis:");
for (int number : numbers) {
    if (isPositive(number)) {
        if (isEven(number)) {
            System.out.println(number + " is positive and even.");
        } else {
            System.out.println(number + " is positive and odd.");
        }
    } else {
        System.out.println(number + " is negative or zero.");
    }
}

int comparisonResult = compare(numbers[0], numbers[numbers.length - 1]);
System.out.println("\nComparison of first and last elements:");

if (comparisonResult == 1) {
    System.out.println("First element is greater than the last element.");
} else if (comparisonResult == 0) {
    System.out.println("First element is equal to the last element.");
} else {
    System.out.println("First element is less than the last element.");
}
```

```
        input.close();
    }
}
```

10.An organization took up the exercise to find the Body Mass Index (BMI) of all the persons in the team of 10 members. For this create a program to find the BMI and display the height, weight, BMI and status of each individual

Hint =>

1. Take user input in double for the weight (in kg) of the person and height (in cm) for the person and and store it in the corresponding 2D array of 10 rows and 3 columns. The First Column storing the weight, the second column storing the height in cm and the third column is the BMI
2. Create a Method to find the BMI of every person and populate the array. Use the formula BMI = weight / (height * height). Note unit is kg/m^2. For this convert cm to meter
3. Create a Method to determine the BMI status using the logic shown in the figure below. and return the array of all the persons BMI Status.

BMI	Status
≤ 18.4	Underweight
18.5 - 24.9	Normal
25.0 - 39.9	Overweight
≥ 40.0	Obese

```
import java.util.Scanner;


public class BMIAnalysis {


    public static double[][] calculateBMI(double[][] personData) {
        for (int i = 0; i < personData.length; i++) {
            double weight = personData[i][0];
            double heightInCm = personData[i][1];
            double heightInMeters = heightInCm / 100.0; // Convert cm to meters
            personData[i][2] = weight / (heightInMeters * heightInMeters);
        }
        return personData;
    }


    public static String[] getBMIStatus(double[][] personData) {
```

```
    String[] statuses = new String[personData.length];
    for (int i = 0; i < personData.length; i++) {
        double bmi = personData[i][2];
        if (bmi < 18.5) {
            statuses[i] = "Underweight";
        } else if (bmi < 25) {
            statuses[i] = "Normal weight";
        } else if (bmi < 30) {
            statuses[i] = "Overweight";
        } else {
            statuses[i] = "Obese";
        }
    }
    return statuses;
}

public static void main(String[] args) {
    Scanner input = new Scanner(System.in);
    double[][] personData = new double[10][3]; // [persons][weight, height(cm), BMI]

    System.out.println("Enter weight (kg) and height (cm) for 10 persons:");

    for (int i = 0; i < 10; i++) {
        boolean validInput = false;
        while(!validInput) {
            System.out.print("Enter weight (kg) for person " + (i + 1) + ": ");
            if (input.hasNextDouble()) {
                personData[i][0] = input.nextDouble();
                validInput = true;
            } else {
                System.out.println("Invalid input. Enter a double");
                input.next();
            }
        }
```

```
            validInput = false;
            while(!validInput) {
                System.out.print("Enter height (cm) for person " + (i + 1) + ": ");
                if (input.hasNextDouble()) {
                    personData[i][1] = input.nextDouble();
                    validInput = true;
                } else {
                    System.out.println("Invalid input. Enter a double");
                    input.next();
                }
            }
        }

        personData = calculateBMI(personData);
        String[] statuses = getBMIStatus(personData);

        System.out.println("\nBMI Analysis:");
        System.out.println("-------------------------------------------------");
        System.out.printf("%-10s %-10s %-10s %-20s\n", "Weight (kg)", "Height (cm)", "BMI", "Weight Status");
        System.out.println("-------------------------------------------------");

        for (int i = 0; i < 10; i++) {
            System.out.printf("%-10.2f %-10.2f %-10.2f %-20s\n", personData[i][0], personData[i][1], personData[i][2], statuses[i]);
        }

        input.close();
    }
}
```

11.Write a program Quadratic to find the roots of the equation ax2+ bx + c. Use Math functions ***Math.pow()*** and ***Math.sqrt()***

Hint =>

1. Take a, b, and c as input values to find the roots of x.

2. The roots are computed using the following formulae
   delta = b2+ 4*a*c
   If delta is positive the find the two roots using formulae
   root1 of x = (-b + delta)/(2*a)
   root1 of x = (-b - delta)/(2*a)
   If delta is zero then there is only one root of x
   root of x = -b/(2*a)
   If delta is negative return empty array or nothing

1. Write a Method to find find the roots of a quadratic equation and return the roots

```
import java.util.Scanner;

public class QuadraticRoots {

    public static double[] findRoots(double a, double b, double c) {
        double delta = Math.pow(b, 2) - 4 * a * c;

        if (delta > 0) {
            double root1 = (-b + Math.sqrt(delta)) / (2 * a);
            double root2 = (-b - Math.sqrt(delta)) / (2 * a);
            return new double[]{root1, root2};
        } else if (delta == 0) {
            double root = -b / (2 * a);
            return new double[]{root};
        } else {
            return new double[0]; // Empty array for negative delta
        }
    }

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

        System.out.print("Enter the value of a: ");
        double a = input.nextDouble();
```

```
        System.out.print("Enter the value of b: ");
        double b = input.nextDouble();

        System.out.print("Enter the value of c: ");
        double c = input.nextDouble();

        double[] roots = findRoots(a, b, c);

        if (roots.length == 2) {
            System.out.println("Roots are: " + roots[0] + " and " + roots[1]);
        } else if (roots.length == 1) {
            System.out.println("Root is: " + roots[0]);
        } else {
            System.out.println("No real roots exist.");
        }

        input.close();
    }
}
```

12.Write a program that generates five 4 digit random values and then finds their average value, and their minimum and maximum value. Use Math.random(), Math.min(), and Math.max().

Hint =>

1. Write a method that generates array of 4 digit random numbers given the size as a parameter as shown in the method signature

   public int[] generate4DigitRandomArray(int size)

2. Write a method to find average, min and max value of an array

public double[] findAverageMinMax(int[] numbers)

```
public class RandomNumberAnalysis {

    public static int[] generate4DigitRandomArray(int size) {
        int[] randomNumbers = new int[size];
        for (int i = 0; i < size; i++) {
```

```
            randomNumbers[i] = 1000 + (int) (Math.random() * 9000); // Generates a number between 1000 and 9999
        }
        return randomNumbers;
    }

    public static double[] findAverageMinMax(int[] numbers) {
        if (numbers == null || numbers.length == 0) {
            return new double[]{0, 0, 0}; // Return default values for empty or null array
        }

        double sum = 0;
        int min = numbers[0];
        int max = numbers[0];

        for (int number : numbers) {
            sum += number;
            min = Math.min(min, number);
            max = Math.max(max, number);
        }

        double average = sum / numbers.length;
        return new double[]{average, min, max};
    }

    public static void main(String[] args) {
        int[] randomValues = generate4DigitRandomArray(5);

        System.out.println("Generated random numbers:");
        for (int value : randomValues) {
            System.out.print(value + " ");
        }
```

```
        System.out.println();

        double[] results = findAverageMinMax(randomValues);
        double average = results[0];
        int min = (int) results[1];
        int max = (int) results[2];

        System.out.println("Average: " + average);
        System.out.println("Minimum: " + min);
        System.out.println("Maximum: " + max);
    }
}
```