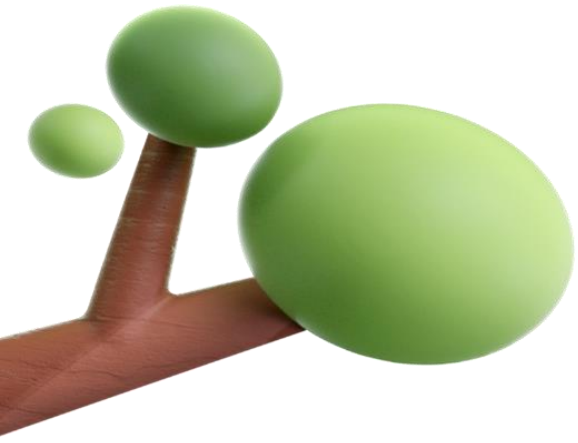


LEVEL 2 ASSIGNMENT

STEP PROGRAM

JAVA METHODS

SUBMITTED BY:

NAME: E.Harikrishna

REG NO: RA2411026010408

CLASS: AJ1

SOURCE CODE

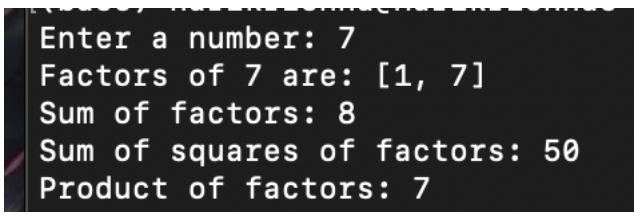
LEVEL 2 QUESTION: 1

Q. 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

```
import java.util.ArrayList;
import java.util.Scanner;
import java.math.BigInteger;
public class FactorCalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = scanner.nextInt();
        ArrayList<Integer> factors = new ArrayList<>();
        int sum = 0;
        int sumOfSquares = 0;
        BigInteger product = BigInteger.ONE;
        for (int i = 1; i <= number; i++) {
            if (number % i == 0) {
```

```
                factors.add(i);
                sum += i;
                sumOfSquares += i * i;
                product = product.multiply(BigInteger.valueOf(i));
            }
        }
        System.out.println("Factors of " + number + " are: " + factors);
        System.out.println("Sum of factors: " + sum);
        System.out.println("Sum of squares of factors: " + sumOfSquares);
        System.out.println("Product of factors: " + product);
        scanner.close();
    }
}
```

OUTPUT:



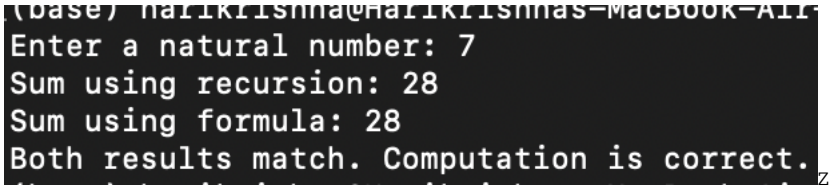
LEVEL2 QUESTION: 2

Q. 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.

```
import java.util.Scanner;
public class NaturalNumberSum {
    public static int recursiveSum(int n) {
        if (n == 1)
            return 1;
        return n + recursiveSum(n - 1);
    }
    public static int formulaSum(int n) {
        return n * (n + 1) / 2;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a natural number: ");
        int n = scanner.nextInt();

        if (n <= 0) {
            System.out.println("Invalid input. Enter a natural number greater than 0.");
```

```
            return;
        }
        int sumRecursive = recursiveSum(n);
        int sumFormula = formulaSum(n);
        System.out.println("Sum using recursion: " + sumRecursive);
        System.out.println("Sum using formula: " + sumFormula);
        if (sumRecursive == sumFormula) {
            System.out.println("Both results match. Computation is correct.");
        } else {
            System.out.println("Results do not match. Check the logic.");
        }
    }
}
```

OUTPUT:



LEVEL 2 QUESTION: 3

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

```
import java.util.Scanner;

public class LeapYearCheck {

  public static boolean isLeapYear(int year) {

    return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);

  }

  public static void main(String[] args) {

    Scanner scanner = new Scanner(System.in);

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

    int year = scanner.nextInt();


    if (year < 1582) {

      System.out.println("Invalid input. Year must be 1582 or later.");

      return;

    }

    if (isLeapYear(year)) {

      System.out.println(year + " is a Leap Year.");

    } else {

      System.out.println(year + " is not a Leap Year.");

    }

  }
```

```
}
```

OUTPUT:

```
Enter a year (>= 1582): 2024
2024 is a Leap Year.
```

LEVEL 2 QUESTION: 4

Q. 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.

```
public class UnitConvertor {
   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) {
        System.out.println("10 kilometers to miles: " + convertKmToMiles(10));
        System.out.println("6.2 miles to kilometers: " + convertMilesToKm(6.2));
        System.out.println("5 meters to feet: " + convertMetersToFeet(5));
        System.out.println("16.4 feet to meters: " + convertFeetToMeters(16.4));
    }
}
```

OUTPUT:

```
10 kilometers to miles: 6.21371
6.2 miles to kilometers: 9.977908000000001
5 meters to feet: 16.4042
16.4 feet to meters: 4.99872
(base) harikrishna@Harikrishnas-MacBook-Air-7 java %
```

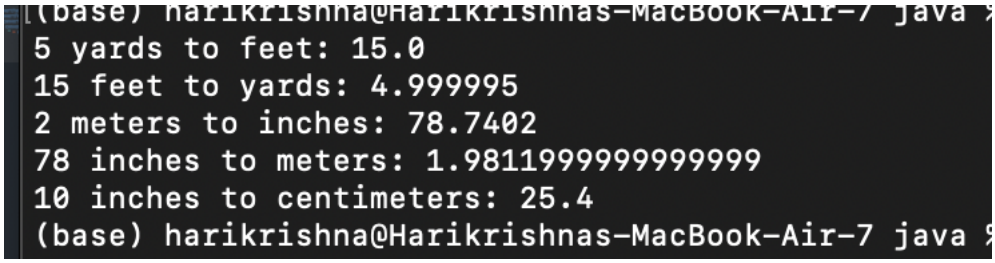
LEVEL 2 QUESTION: 5

Q. 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.

```
public class CovertorUnit{
   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) {
        System.out.println("5 yards to feet: " + convertYardsToFeet(5));
        System.out.println("15 feet to yards: " + convertFeetToYards(15));
        System.out.println("2 meters to inches: " + convertMetersToInches(2));
        System.out.println("78 inches to meters: " + convertInchesToMeters(78));
        System.out.println("10 inches to centimeters: " + convertInchesToCentimeters(10));
    }
}
```

OUTPUT:



LEVEL 2 QUESTION: 6

Q. 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

```
public class ConvertorUnit {
   public static double convertFarhenheitToCelsius(double farhenheit) {
      return (farhenheit - 32) * 5 / 9;
   }
   public static double convertCelsiusToFarhenheit(double celsius) {
      return (celsius * 9 / 5) + 32;
   }
   public static double convertPoundsToKilograms(double pounds) {
      double pounds2kilograms = 0.453592;
      return pounds * pounds2kilograms;
   }
   public static double convertKilogramsToPounds(double kilograms) {
      double kilograms2pounds = 2.20462;
      return kilograms * kilograms2pounds;
   }
   public static double convertGallonsToLiters(double gallons) {
      double gallons2liters = 3.78541;
      return gallons * gallons2liters;
```

```
    }
    public static double convertLitersToGallons(double liters) {
        double liters2gallons = 0.264172;
        return liters * liters2gallons;
    }
    public static void main(String[] args) {
        System.out.println("98.6°F to Celsius: " + convertFarhenheitToCelsius(98.6));
        System.out.println("37°C to Fahrenheit: " + convertCelsiusToFarhenheit(37));
        System.out.println("150 pounds to kilograms: " +
convertPoundsToKilograms(150));
        System.out.println("68 kilograms to pounds: " + convertKilogramsToPounds(68));
        System.out.println("10 gallons to liters: " + convertGallonsToLiters(10));
        System.out.println("20 liters to gallons: " + convertLitersToGallons(20));
    }
}
```

OUTPUT:

```
98.6°F to Celsius: 37.0
37°C to Fahrenheit: 98.6
150 pounds to kilograms: 68.0388
68 kilograms to pounds: 149.91415999999998
10 gallons to liters: 37.8541
20 liters to gallons: 5.283440000000001
```

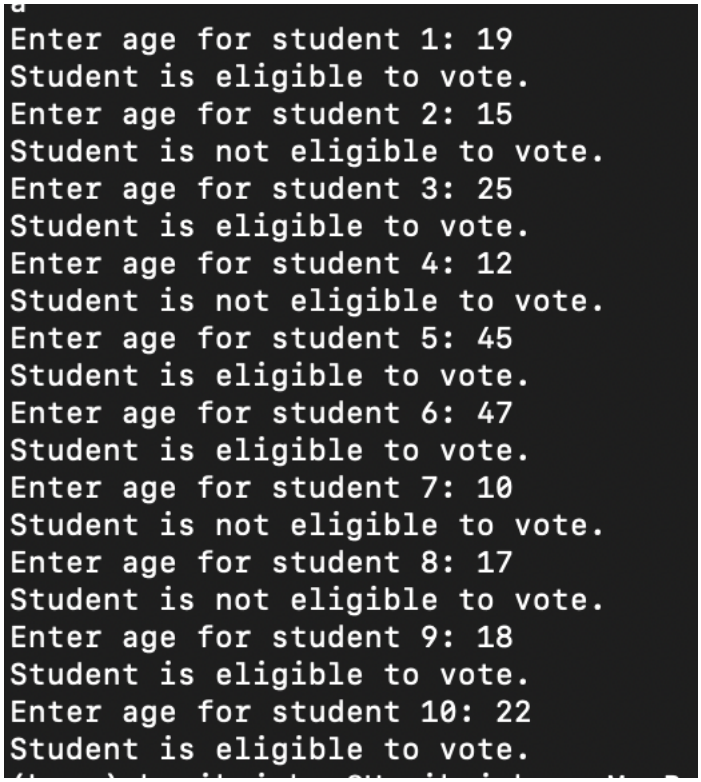
LEVEL 2 QUESTION: 7

Q. 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.

```
import java.util.Scanner;

public class StudentVoteChecker {

   public static boolean canStudentVote(int age) {

      if (age < 0) {

         return false;

      } else if (age >= 18) {

         return true;

      } else {

         return false;

      }

   }

   public static void main(String[] args) {

      Scanner scanner = new Scanner(System.in);

      int[] ages = new int[10];

      for (int i = 0; i < ages.length; i++) {

         System.out.print("Enter age for student " + (i + 1) + ": ");

         ages[i] = scanner.nextInt();

         boolean canVote = canStudentVote(ages[i]);
```

```
        if (ages[i] < 0) {
            System.out.println("Invalid age. Cannot vote.");
        } else if (canVote) {
            System.out.println("Student is eligible to vote.");
        } else {
            System.out.println("Student is not eligible to vote.");
        }
    }
  }
}
```

OUTPUT:



LEVEL 2 QUESTION: 8

Q. 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

```
import java.util.Scanner;

public class FriendsInfo {

   public static int findYoungestIndex(int[] ages) {

      int minIndex = 0;

      for (int i = 1; i < ages.length; i++) {

         if (ages[i] < ages[minIndex]) {

            minIndex = i;

         }

      }

      return minIndex;

   }

   public static int findTallestIndex(double[] heights) {
```

```
    int maxIndex = 0;
    for (int i = 1; i < heights.length; i++) {
        if (heights[i] > heights[maxIndex]) {
            maxIndex = i;
        }
    }
    return maxIndex;
}
public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    String[] names = {"Amar", "Akbar", "Anthony"};
    int[] ages = new int[3];
    double[] heights = new double[3];

    for (int i = 0; i < 3; i++) {
        System.out.print("Enter age of " + names[i] + ": ");
        ages[i] = scanner.nextInt();
```

```
        System.out.print("Enter height (in cm) of " + names[i] + ": ");
        heights[i] = scanner.nextDouble();
    }
    int youngestIndex = findYoungestIndex(ages);
    int tallestIndex = findTallestIndex(heights);

    System.out.println("The youngest friend is: " + names[youngestIndex]);
    System.out.println("The tallest friend is: " + names[tallestIndex]);
  }
}
```

OUTPUT:

```
Enter age of Amar: 18
Enter height (in cm) of Amar: 173
Enter age of Akbar: 19
Enter height (in cm) of Akbar: 168
Enter age of Anthony: 19
Enter height (in cm) of Anthony: 175
The youngest friend is: Amar
The tallest friend is: Anthony
```

LEVEL 2 QUESTION: 9

Q. 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

```
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 scanner = new Scanner(System.in);
```

```
        int[] numbers = new int[5];
        for (int i = 0; i < numbers.length; i++) {
            System.out.print("Enter number " + (i + 1) + ": ");
            numbers[i] = scanner.nextInt();
            if (isPositive(numbers[i])) {
                System.out.print("The number is positive and ");
                if (isEven(numbers[i])) {
                    System.out.println("even.");
                } else {
                    System.out.println("odd.");
                }
            } else {
                System.out.println("The number is negative.");
            }
        }
        int result = compare(numbers[0],
numbers[numbers.length - 1]);
        if (result == 1) {
```

```
        System.out.println("The first number is greater than the last number.");
    } else if (result == 0) {
        System.out.println("The first number is equal to the last number.");
    } else {
        System.out.println("The first number is less than the last number.");
    }
  }
}
```

OUTPUT:

```
Enter number 1: 10
The number is positive and even.
Enter number 2: 15
The number is positive and odd.
Enter number 3: 5
The number is positive and odd.
Enter number 4: 20
The number is positive and even.
Enter number 5: 25
The number is positive and odd.
The first number is less than the last number.
```

LEVEL 2 QUESTION: 10

Q. 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

```
import java.util.Scanner;

public class BMICalc{
   public static void calculateBMI(double[][] data) {
      for (int i = 0; i < data.length; i++) {
         double weight = data[i][0];
         double heightCm = data[i][1];
         double heightM = heightCm / 100.0;
         double bmi = weight / (heightM * heightM);
         data[i][2] = bmi;
      }
   }
   public static String[] getBMIStatus(double[][] data) {
      String[] status = new String[data.length];
      for (int i = 0; i < data.length; i++) {
         double bmi = data[i][2];
         if (bmi < 18.5) {
```

```
            status[i] = "Underweight";
        } else if (bmi >= 18.5 && bmi < 24.9) {
            status[i] = "Normal weight";
        } else if (bmi >= 25 && bmi < 29.9) {
            status[i] = "Overweight";
        } else {
            status[i] = "Obese";
        }
    }
    return status;
  }
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    double[][] teamData = new double[10][3]; // 0: weight, 1: height, 2: BMI

    for (int i = 0; i < teamData.length; i++) {
        System.out.print("Enter weight (kg) of person " + (i + 1) + ": ");
```

```
            teamData[i][0] = scanner.nextDouble();
            System.out.print("Enter height (cm) of person " + (i + 1) + ": ");
            teamData[i][1] = scanner.nextDouble();
        }
        calculateBMI(teamData);
        String[] status = getBMIStatus(teamData);
        System.out.println("\n--- BMI Report ---");
        for (int i = 0; i < teamData.length; i++) {
            System.out.printf("Person %d - Weight: %.2f kg, Height: %.2f cm, BMI: %.2f, Status: %s\n",
                    (i + 1), teamData[i][0], teamData[i][1], teamData[i][2], status[i]);
        }
    }
}
```

OUTPUT:

```
Enter weight (kg) of person 1: 75
Enter height (cm) of person 1: 168
Enter weight (kg) of person 2: 80
Enter height (cm) of person 2: 173
Enter weight (kg) of person 3: 90
Enter height (cm) of person 3: 175
Enter weight (kg) of person 4: 68
Enter height (cm) of person 4: 168
Enter weight (kg) of person 5: 95
Enter height (cm) of person 5: 172
Enter weight (kg) of person 6: 96
Enter height (cm) of person 6: 171
Enter weight (kg) of person 7: 83
Enter height (cm) of person 7: 170
Enter weight (kg) of person 8: 84
Enter height (cm) of person 8: 174
Enter weight (kg) of person 9: 66
Enter height (cm) of person 9: 165
Enter weight (kg) of person 10: 96
Enter height (cm) of person 10: 176

--- BMI Report ---
Person 1 - Weight: 75.00 kg, Height: 168.00 cm, BMI: 26.57, Status: Overweight
Person 2 - Weight: 80.00 kg, Height: 173.00 cm, BMI: 26.73, Status: Overweight
Person 3 - Weight: 90.00 kg, Height: 175.00 cm, BMI: 29.39, Status: Overweight
Person 4 - Weight: 68.00 kg, Height: 168.00 cm, BMI: 24.09, Status: Normal weight
Person 5 - Weight: 95.00 kg, Height: 172.00 cm, BMI: 32.11, Status: Obese
Person 6 - Weight: 96.00 kg, Height: 171.00 cm, BMI: 32.83, Status: Obese
Person 7 - Weight: 83.00 kg, Height: 170.00 cm, BMI: 28.72, Status: Overweight
Person 8 - Weight: 84.00 kg, Height: 174.00 cm, BMI: 27.74, Status: Overweight
Person 9 - Weight: 66.00 kg, Height: 165.00 cm, BMI: 24.24, Status: Normal weight
Person 10 - Weight: 96.00 kg, Height: 176.00 cm, BMI: 30.99, Status: Obese
```

LEVEL 2 QUESTION: 11

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

import java.util.Scanner; public class Quadratic { 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[]{}; // No real roots
    }
}
public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter coefficient a: ");
    double a = scanner.nextDouble();
    System.out.print("Enter coefficient b: ");
    double b = scanner.nextDouble();
    System.out.print("Enter coefficient c: ");
    double c = scanner.nextDouble();
    if (a == 0) {
        System.out.println("This is not a quadratic equation.");
        return;
    }
    double[] roots = findRoots(a, b, c);
    if (roots.length == 2) {
        System.out.println("Two distinct roots: x1 = " + roots[0] + ",
```

```
x2 = " + roots[1]);
  } else if (roots.length == 1) {
    System.out.println("One real root: x = " + roots[0]);
  } else {
    System.out.println("No real roots.");
  }
}


}
```

OUTPUT:

```
Enter coefficient a: 3
Enter coefficient b: 4
Enter coefficient c: 5
No real roots.
```

LEVEL 2 QUESTION: 12

Q. 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().

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

```
public class RandomStats {
    public static int[] generate4DigitRandomArray(int size) {
        int[] numbers = new int[size];
        for (int i = 0; i < size; i++) {
            numbers[i] = (int)(Math.random() * 9000) + 1000; // 1000
to 9999
        }
        return numbers;
    }
    public static double[] findAverageMinMax(int[] numbers) {
        int min = numbers[0];
        int max = numbers[0];
        int sum = 0;
        for (int number : numbers) {
            sum += number;
            min = Math.min(min, number);
            max = Math.max(max, number);
        }
```

```
        double average = (double) sum / numbers.length;
        return new double[]{average, min, max};
    }
    public static void main(String[] args) {
        int[] randomNumbers = generate4DigitRandomArray(5);
        System.out.println("Generated 4-digit numbers: " +
Arrays.toString(randomNumbers));

        double[] stats = findAverageMinMax(randomNumbers);
        System.out.printf("Average: %.2f, Min: %.0f, Max: %.0f\n",
stats[0], stats[1], stats[2]);
    }
}
```

OUTPUT: