Q1. Factor Calculator

```
import java.util.Scanner;

public class FactorCalculator {
    public static int[] findFactors(int num) {
        int count = 0;
        for (int i = 1; i <= num; i++) if (num % i == 0) count++;
        int[] factors = new int[count];
        int idx = 0;
        for (int i = 1; i <= num; i++) if (num % i == 0) factors[idx++] = i;
        return factors;
    }

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

    public static int productFactors(int[] factors) {
        int product = 1;
        for (int f : factors) product *= f;
        return product;
    }

    public static double sumSquares(int[] factors) {
        double sum = 0;
        for (int f : factors) sum += Math.pow(f, 2);
        return sum;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        int[] factors = findFactors(num);
        System.out.println("Factors:");
        for (int f : factors) System.out.print(f + " ");
        System.out.println("\nSum: " + sumFactors(factors));
        System.out.println("Product: " + productFactors(factors));
        System.out.println("Sum of Squares: " + sumSquares(factors));
    }
}
```

Q2. Natural Sum

```
import java.util.Scanner;

public class NaturalSum {
    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 sc = new Scanner(System.in);
        System.out.print("Enter a natural number: ");
        int n = sc.nextInt();
        if (n <= 0) {
            System.out.println("Not a natural number.");
            return;
        }
        System.out.println("Recursive Sum: " + recursiveSum(n));
        System.out.println("Formula Sum: " + formulaSum(n));
    }
}
```

Q3. Leap Year

```
import java.util.Scanner;

public class LeapYear {
    public static boolean isLeap(int year) {
        return year >= 1582 && (year % 4 == 0 && year % 100 != 0 || year % 400 == 0);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a year: ");
        int year = sc.nextInt();
        if (isLeap(year)) System.out.println("Leap Year");
        else System.out.println("Not a Leap Year");
    }
}
```

Q4. Km to Miles

```
public class UnitConverter {
    public static double convertKmToMiles(double km) {
        return km * 0.621371;
    }

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

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

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

Q5. Yards to Feet

```
public class ExtendedUnitConverter {
    public static double convertYardsToFeet(double yards) {
        return yards * 3;
    }

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

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

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

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

Q6. Fahrenheit to Celsius

```
public class MoreUnitConverter {
```

```
    public static double convertFahrenheitToCelsius(double f) {
        return (f - 32) * 5 / 9;
    }

    public static double convertCelsiusToFahrenheit(double c) {
        return (c * 9 / 5) + 32;
    }

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

    public static double convertKgToPounds(double kg) {
        return kg * 2.20462;
    }

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

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

Q7. Student Vote Checker

```
import java.util.Scanner;

public class StudentVoteChecker {
    public static boolean canStudentVote(int age) {
        if (age < 0) return false;
        return age >= 18;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] ages = new int[10];
        for (int i = 0; i < 10; i++) {
            System.out.print("Enter age of student " + (i + 1) + ": ");
            ages[i] = sc.nextInt();
            if (canStudentVote(ages[i])) System.out.println("Can Vote");
            else System.out.println("Cannot Vote");
        }
    }
}
```

Q8. Friend Comparison

```
import java.util.Scanner;

public class FriendComparison {
    public static int findYoungest(int[] ages) {
        int min = ages[0], idx = 0;
        for (int i = 1; i < ages.length; i++) if (ages[i] < min) {
            min = ages[i]; idx = i;
        }
        return idx;
    }

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

    public static void main(String[] args) {
        Scanner sc = 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] = sc.nextInt();
            System.out.print("Enter height of " + names[i] + ": ");
            heights[i] = sc.nextDouble();
        }

        System.out.println("Youngest: " + names[findYoungest(ages)]);
        System.out.println("Tallest: " + names[findTallest(heights)]);
    }
}
```

Q9. Number Analysis

```
import java.util.Scanner;

public class NumberAnalysis {
    public static boolean isPositive(int n) {
        return n >= 0;
    }
```

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

    public static int compare(int a, int b) {
        return Integer.compare(a, b); // -1 if a < b, 0 if a == b, 1 if a > b
    }

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

        for (int i = 0; i < 5; i++) {
            System.out.print("Enter number " + (i + 1) + ": ");
            numbers[i] = sc.nextInt();

            if (isPositive(numbers[i])) {
                System.out.print("Positive ");
                System.out.println(isEven(numbers[i]) ? "Even" : "Odd");
            } else {
                System.out.println("Negative");
            }
        }

        int cmp = compare(numbers[0], numbers[4]);
        if (cmp == 0) System.out.println("First and last are Equal");
        else if (cmp > 0) System.out.println("First is Greater");
        else System.out.println("First is Lesser");
    }
}
```

Q10. BMI Calculator

```
import java.util.Scanner;

public class BMICalculator {
    public static double calculateBMI(double weight, double heightCm) {
        double heightM = heightCm / 100.0;
        return weight / (heightM * heightM);
    }

    public static String getStatus(double bmi) {
        if (bmi < 18.5) return "Underweight";
        else if (bmi < 24.9) return "Normal";
        else if (bmi < 29.9) return "Overweight";
        else return "Obese";
```

```
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double[][] data = new double[10][3];

        for (int i = 0; i < 10; i++) {
            System.out.print("Enter weight(kg) for person " + (i + 1) + ": ");
            data[i][0] = sc.nextDouble();
            System.out.print("Enter height(cm) for person " + (i + 1) + ": ");
            data[i][1] = sc.nextDouble();
            data[i][2] = calculateBMI(data[i][0], data[i][1]);
        }

        for (int i = 0; i < 10; i++) {
            System.out.printf("Person %d - Height: %.1fcm, Weight: %.1fkg, BMI: %.2f, Status:
%s\n",
                    (i + 1), data[i][1], data[i][0], data[i][2], getStatus(data[i][2]));
        }
    }
}
```

Q11. Quadratic Solver

```
import java.util.Scanner;

public class QuadraticSolver {
    public static double[] findRoots(double a, double b, double c) {
        double delta = Math.pow(b, 2) - 4 * a * c;
        if (delta < 0) return new double[0];
        else if (delta == 0) return new double[]{ -b / (2 * a) };
        else {
            double root1 = (-b + Math.sqrt(delta)) / (2 * a);
            double root2 = (-b - Math.sqrt(delta)) / (2 * a);
            return new double[]{ root1, root2 };
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a, b, c: ");
        double a = sc.nextDouble(), b = sc.nextDouble(), c = sc.nextDouble();

        double[] roots = findRoots(a, b, c);
        if (roots.length == 0) System.out.println("No Real Roots");
        else for (double r : roots) System.out.println("Root: " + r);
    }
```

```
}
```

Q12. Random Array Stats

```
import java.util.Random;

public class RandomArrayStats {
    public static int[] generate4DigitRandomArray(int size) {
        Random rand = new Random();
        int[] arr = new int[size];
        for (int i = 0; i < size; i++)
            arr[i] = 1000 + rand.nextInt(9000);
        return arr;
    }

    public static double[] findAverageMinMax(int[] arr) {
        int min = arr[0], max = arr[0], sum = 0;
        for (int n : arr) {
            if (n < min) min = n;
            if (n > max) max = n;
            sum += n;
        }
        return new double[]{ (double) sum / arr.length, min, max };
    }

    public static void main(String[] args) {
        int[] arr = generate4DigitRandomArray(5);
        System.out.print("Array: ");
        for (int n : arr) System.out.print(n + " ");
        System.out.println();

        double[] stats = findAverageMinMax(arr);
        System.out.println("Average: " + stats[0]);
        System.out.println("Min: " + stats[1]);
        System.out.println("Max: " + stats[2]);
    }
}
```