Nama	:	Teosofi Hidayah Agung
NRP	:	5002221132

11.4 (*Maximum element in* ArrayList) Write the following method that returns the maximum value in an ArrayList of integers. The method returns null if the list is null or the list size is 0.

```
public static Integer max(ArrayList<Integer> list)
```

Solve:
The method can be implemented as follows:

```
public static Integer max(ArrayList<Integer> list) {
    if (list == null || list.size() == 0) return null;
    Integer max = list.get(0);
    for (int i = 1; i < list.size(); i++) {
        if (list.get(i) > max) max = list.get(i);
    }
    return max;
}
```

`Integer` is used instead of `int` to allow the method to return `null` if the list is null or empty.

11.9 (*Largest row and column*) Write a program that randomly fills in 0s and 1s into a n-by-n matrix, prints the matrix, and finds the rows and columns with the most 1s. (Hint: Use two ArrayLists to store the row and column indices with the most 1s.) Here is a sample run of the program:

```
Enter the array size n: 4 ↵
The random array is
0011
0011
1101
1010
The largest row index: 2
The largest column index: 2, 3
```

Solve:
Index of row and coloumn it's start from zero. First, we create all the necessary method as follows:

```
package Aplro2.Week14;

import java.util.ArrayList;
import java.util.Random;

public class Liang_Ch11_9 {
    // Create a n-by-n matrix filled with random 0s and 1s
    public static ArrayList<ArrayList<Integer>> rMatrix(int size) {
        Random random = new Random();
        ArrayList<ArrayList<Integer>> list = new ArrayList<>();
        for (int i = 0; i < size; i++) {
            ArrayList<Integer> innerList = new ArrayList<>();
            for (int j = 0; j < size; j++) {
                innerList.add(random.nextInt(2));
            }
            list.add(innerList);
        }
        return list;
    }
    // Return the Row index with the most 1s
    public static ArrayList<ArrayList<Integer>>
       MaxRowCol(ArrayList<ArrayList<Integer>> list) {
        ArrayList<ArrayList<Integer>> result = new ArrayList<>();
        ArrayList<Integer> maxRow = new ArrayList<>();
        ArrayList<Integer> maxCol = new ArrayList<>();
        int maxR = 0;
        int maxC = 0;
        for (int i = 0; i < list.size(); i++) {
            int sumR = 0;
            int sumC = 0;
            for (int j = 0; j < list.get(i).size(); j++) {
                sumR += list.get(i).get(j);
                sumC += list.get(j).get(i);
            }
            //Check Row
            if (sumR > maxR) {
                maxR = sumR;
                maxRow.clear();
                maxRow.add(i);
            } else if (sumR == maxR) {
                maxRow.add(i);
            }
            //Check Coloumn
            if (sumC > maxC) {
                maxC = sumC;
                maxCol.clear();
                maxCol.add(i);
            } else if (sumC == maxC) {
                maxCol.add(i);
            }
        }
        result.add(maxRow);
        result.add(maxCol);
        return result;
    }
    // Print the matrix
    public static void PrintMatrix(ArrayList<ArrayList<Integer>> list) {
        for (int i = 0; i < list.size(); i++) {
            for (int j = 0; j < list.get(i).size(); j++) {
                System.out.print(list.get(i).get(j));
            }
            System.out.println();
        }
    }
    // ArrayList to String
    public static String PrintArrayList(ArrayList<Integer> list) {
        String result = "";
        for (int i = 0; i < list.size(); i++) {
            if (i == list.size() - 1){
                result += list.get(i);
                break;
            }
            result += list.get(i) + ", ";
        }
        return result;
    }
}
```

Little explanation about the code above:

- `rMatrix` method is used to create a matrix with random 0s and 1s.
- `MaxRowCol` method is used to find the row and column with the most 1s.
- `PrintMatrix` method is used to print the matrix.
- `PrintArrayList` method is used to convert `ArrayList` to `String`.

Then, we create the main program as follows:

```
package Aplro2.Week14;

import java.util.Scanner;
import java.util.ArrayList;

public class CheckMethod extends Liang_Ch11_9{
    public static void main(String[] args) {
        System.out.print("Enter the array size n: ");
        Scanner input = new Scanner(System.in);
        int n = input.nextInt();
        ArrayList<ArrayList<Integer>> list = rMatrix(n);
        PrintMatrix(list);
        ArrayList<ArrayList<Integer>> result = MaxRowCol(list);
        System.out.println("The largest row index: " +
            PrintArrayList(result.get(0)));
        System.out.println("The largest column index: " +
            PrintArrayList(result.get(1)));
    }
}
```

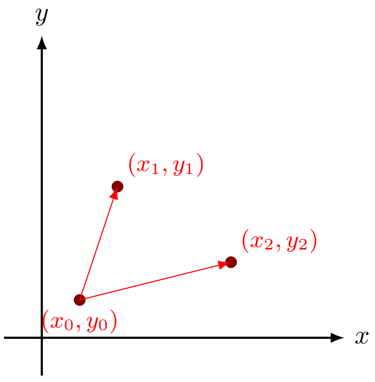
Here is another sample run of the program:

```
Enter the array size n: 5 ↵
01010
11000
00010
10001
11001
The largest row index: 4
The largest column index: 0, 1
```

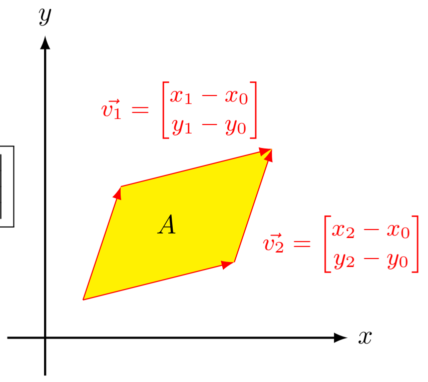
11.15 (*Area of a convex polygon*) A polygon is convex if it contains any line segments that connects two points of the polygon. Write a program that prompts the user to enter the number of points in a convex polygon, then enter the points clockwise, and display the area of the polygon. Here is a sample run:

```
Enter the number of the points: 7 ↵
Enter the coordinates of the points:
   -12 0 -8.5 10 0 11.4 5.5 7.8 6 -5.5 0 -7 -3.5 -3.5 ↵
The total area 250.075
```

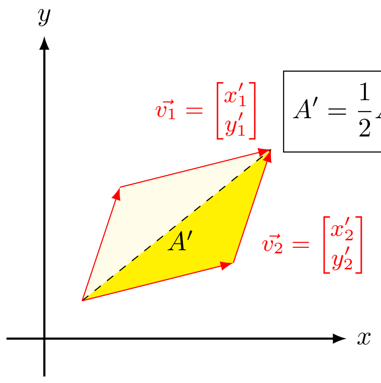
Solve:
First, we need learn about usefull formula from Linear Algebra. Determinant of a 2×2 matrix represented area of area beetween two vector in coloumn of matrix.



$$\boxed{A = \begin{vmatrix} \vec{v_1} & \vec{v_2} \end{vmatrix} = \begin{vmatrix} x'_1 & x'_2 \\ y'_1 & y'_2 \end{vmatrix}}$$

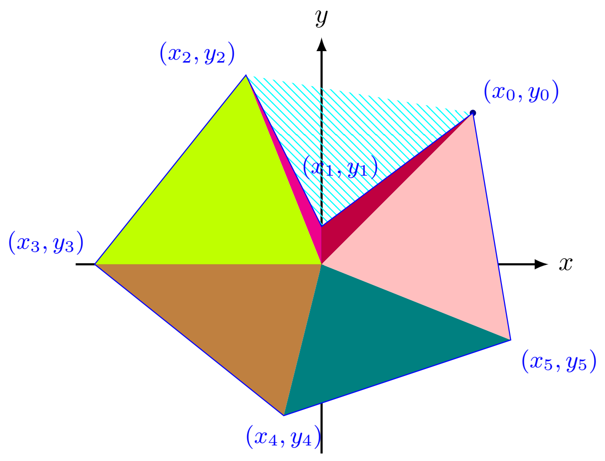

y
$\vec{v_1} = \begin{bmatrix} x_1 - x_0 \\ y_1 - y_0 \end{bmatrix}$
A
$\vec{v_2} = \begin{bmatrix} x_2 - x_0 \\ y_2 - y_0 \end{bmatrix}$
x

Now divide the parallelogram into two triangle such that we get the area of triangle is half of the area of parallelogram.


y
$\vec{v_1} = \begin{bmatrix} x'_1 \\ y'_1 \end{bmatrix}$
A'
$\vec{v_2} = \begin{bmatrix} x'_2 \\ y'_2 \end{bmatrix}$
x

$$\boxed{A' = \frac{1}{2}A = \frac{1}{2}\begin{vmatrix} x'_1 & x'_2 \\ y'_1 & y'_2 \end{vmatrix}}$$

The area of a convex polygon can be calculated by summing the area of all triangles formed by the first point and two consecutive points. The illustration is shown below:



In previous illustration, we can see that the area of the polygon is the sum of the area of all triangles. If the polygon isn't convex, we can count the area like cyan triangle and sum the area of all triangles. We can generalize the formula for the area of a triangle with the following formula:

$$A = \frac{1}{2}\sum_{k=1}^{n}\begin{vmatrix} x_{k-1} & x_{(k \mod n)} \\ y_{k-1} & y_{(k \mod n)} \end{vmatrix}$$

The program can be implemented as follows:

```java
package Aplro2.Week14;

import java.util.ArrayList;
import java.util.Scanner;

public class Liang_Ch11_15 {
    public static double DetArea2D(ArrayList<Double> vec1,
        ArrayList<Double> vec2) {
        return vec1.get(0) * vec2.get(1) - vec1.get(1) * vec2.get(0);
    }
    public static ArrayList<Double> Vector(ArrayList<Double> origin,
        ArrayList<Double> end) {
        ArrayList<Double> vector = new ArrayList<>();
        vector.add(end.get(0) - origin.get(0));
        vector.add(end.get(1) - origin.get(1));
        return vector;
    }
    public static ArrayList<ArrayList<Double>> InputPoints(int n){
        ArrayList<ArrayList<Double>> points = new ArrayList<>();
        try (Scanner input = new Scanner(System.in)) {
            for (int i = 0; i < n; i++) {
                ArrayList<Double> innerList = new ArrayList<>();
                for (int j = 0; j < 2; j++) {
                    innerList.add(input.nextDouble());
                }
                points.add(innerList);
            }
        }
        return points;
    }
    public static double AreaPolygon(ArrayList<ArrayList<Double>>
        points) {
        double area = 0;
        for (int i = 1; i < points.size()-1; i++) {
            ArrayList<Double> vec1 = Vector(points.get(0),
                points.get(i));
            ArrayList<Double> vec2 = Vector(points.get(0), points.get(i
                + 1));
            area += Math.abs(DetArea2D(vec1, vec2));
        }
        return area / 2;
    }
}
```

Some explanation about the code above:

- `DetArea2D` method is used to calculate the determinant of a 2×2 matrix.
- `Vector` method is used to calculate the vector between two points.
- `InputPoints` method is used to input the points of the polygon.

- `AreaPolygon` method is used to calculate the area of the polygon.

Eventually, we create the main program as follows:

```java
package Aplro2.Week14;
import java.util.Scanner;

public class CheckMethod extends Liang_Ch11_15{
    public static void main(String[] args) {
        try (Scanner input = new Scanner(System.in)) {
            System.out.print("Enter the number of the points: ");
            int n = input.nextInt();
            System.out.println("Enter the coordinates of the points: ");
            System.out.println("Total area is: " +
                AreaPolygon(InputPoints(n)));
        }
    }
}
```

Here is another sample run of the program:

```
Enter the number of the points: 6 ↵
Enter the coordinates of the points:
4 4 5 -2 -1 -4 -6 0 -2 5 0 1 ↵
Total area is: 77.0
```

11.16 (*Addition quiz*) Rewrite Listing 5.1 RepeatAdditionQuiz.java to alert the user if an answer is entered again. *Hint: Use an array list to store answers.* Here is a sample run:

```
What is 5 + 9? 12 ↵
Wrong answer. Try again. What is 5 + 9? 34 ↵
Wrong answer. Try again. What is 5 + 9? 12 ↵
You already entered 12
Wrong answer. Try again. What is 5 + 9? 14 ↵
You got it!
```

Solve:
The program can be implemented as follows:

```java
package Aplro2.Week14;

import java.util.ArrayList;
import java.util.Scanner;

public class Liang_Ch11_16 {
    public static void main(String[] args) {
        try (Scanner input = new Scanner(System.in)) {
            int number1 = (int) (Math.random() * 10);
            int number2 = (int) (Math.random() * 10);
            int answer;
            ArrayList<Integer> answers = new ArrayList<>();

            System.out.print("What is " + number1 + " + " + number2 + "?
                ");
            answer = input.nextInt();

            while (answer != number1 + number2) {
                if (answers.contains(answer)) {
                    System.out.println("You already entered " + answer);
                } else {
                    System.out.println("Wrong answer. Try again.");
                    answers.add(answer);
                }
                System.out.print("What is " + number1 + " + " + number2
                    + "? ");
                answer = input.nextInt();
            }
        }
        System.out.println("You got it!");
    }
}
```

Then we can run the program. Here is a sample run of the program:

```
What is 7 + 8? 9 ↵
Wrong answer. Try again.
What is 7 + 8? 0 ↵
Wrong answer. Try again.
What is 7 + 8? 9 ↵
You already entered 9
What is 7 + 8? 0 ↵
You already entered 0
What is 7 + 8? 15 ↵
You got it!
```