Exersize 1.

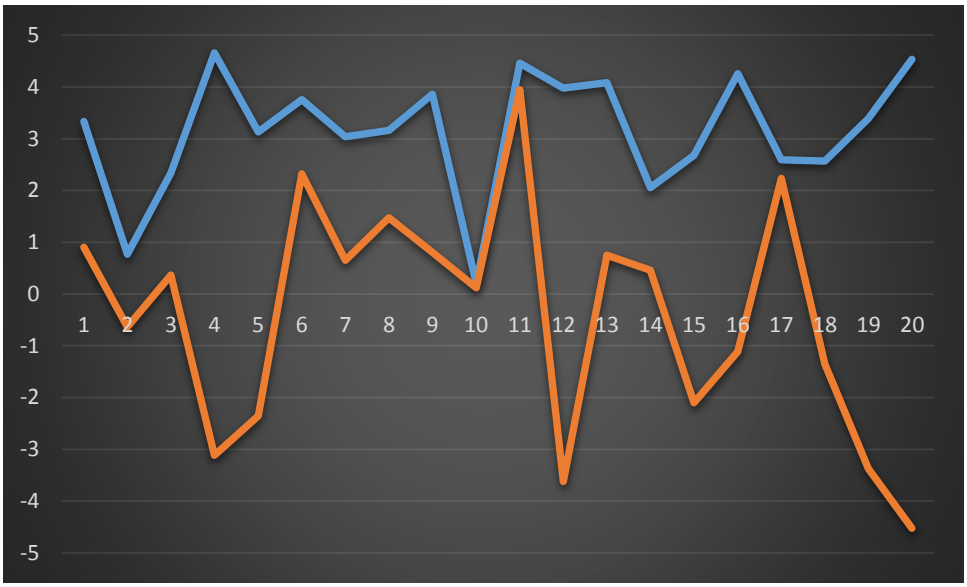
```
import java.util.Arrays;
import java.util.Random;
import java.math.*;

public class siitT {
	public static int chromosoms = 20;
	public static int genes = 20;
	public static float tmp = 0.0f;
	public static float[][] population = new float[chromosoms][genes];
	public static float[] resultY = new float[chromosoms];
	public static float sum = 0f;
	public static float lastY = 0;
	public static float[] best = new float[genes];
	public static float []xz = new float[chromosoms];


	public static void generatePop(){
		Random rnd = new Random();
		for (int i = 0; i < chromosoms; i++) {
			for (int j = 0; j < genes; j++) {
				population[i][j] = rnd.nextInt(1000)/100f - 5;
			}
		}
	}
	public static void main(String[] args) {

		generatePop();
		for (int i = 0; i < chromosoms; i++) {
			for (int j = 0; j < genes; j++) {
			System.out.print(population[i][j] + " ");
			}
			System.out.println();
		}

		for (int i = 0; i < population.length; i++) {
			for (int j = 0; j < genes; j++) {
				float a = (float) (population[i][j]*Math.sin(
Math.abs(population[i][j])));
				sum =+ a;
			}
			resultY[i] = (float)sum;
		}

		for (int i = 0; i < resultY.length; i++) {
			xz[i] = resultY[i];
		}
		Arrays.sort(xz);
		lastY = xz[9];

		for (int i = 0; i < resultY.length; i++) {
			if (resultY[i] <= lastY) {
				best[i] = resultY[i];
			}
			}
		System.out.println("*************");
		for (int i = 0; i < resultY.length; i++) {
			System.out.print(resultY[i] + " ");
		}
		System.out.println();
		for (int i = 0; i < best.length; i++) {
			System.out.print(best[i] + " ");
		}
	}}
```

Exersize 2.

```
package siitTT;

import java.util.Random;

public class siitTT {

	public static int chromososm = 20;
	public static int genes = 10;
	public static int[][] population = new int[chromososm][genes];
	public static double numb = 0d;
	public static double tmp = 0d;

	public static double converte(int[] k) {
		double numb1 = 0;
		for (int i = 9; i >= 0; i--) {
			double tmp1 = k[i] * Math.pow(2, i);
			numb1 = numb1 + tmp1;
		}
		return numb1/1023*5;
	}

	public static double from2To10(int[] k){
		for (int i = 9; i >= 0; i--) {
			tmp = k[i] * Math.pow(2, i);
			numb = numb + tmp;
		}
		numb = numb * Math.sin(Math.abs(numb))/1023*5;
		if(k[0] == 0)
			return -(numb);
		else {
			return numb;
		}
	}


	public static int fitness(int[] a){
		int f = 0;
		for (int i = 0; i < a.length; i++) {
			f =+ a[i];
		}
		return f;
	}
	public static void generatePop(){
		Random rnd = new Random();
		for (int i = 0; i < chromososm; i++) {
			for (int j = 0; j < genes; j++) {
				population[i][j] = rnd.nextInt(2);
			}
		}
	}

	public static void main(String[] args) {

		generatePop();
		for (int i = 0; i < chromososm; i++) {
			for (int j = 0; j < genes; j++) {
				System.out.print(population[i][j] + " ");
			}
			System.out.println(" ");
		}

		for (int i = 0; i < population.length; i++) {
			//System.out.print(fitness(population[i]) + " " );
```

```
                System.out.println(converte(population[i]) + " " +
from2To10(population[i]) + " " );
            }
        }

}
```

Start population (Y and X)



3,333333333	0,90321274
0,767350929	-0,634823333
2,336265885	0,364588558
4,657869013	-3,114835211
3,128054741	-2,355261923
3,753665689	2,323520081
3,035190616	0,651840032
3,16226784	1,472268863
3,861192571	0,80292219
0,200391007	0,11789973
4,462365591	3,947766146
3,978494624	-3,621895762
4,08113392	0,747288786
2,052785924	0,461062969
2,678396872	-2,099311468
4,257086999	-1,111257507
2,595307918	2,234065641
2,565982405	-1,354594692
3,39198436	-3,375649235
4,535679374	-4,526351449