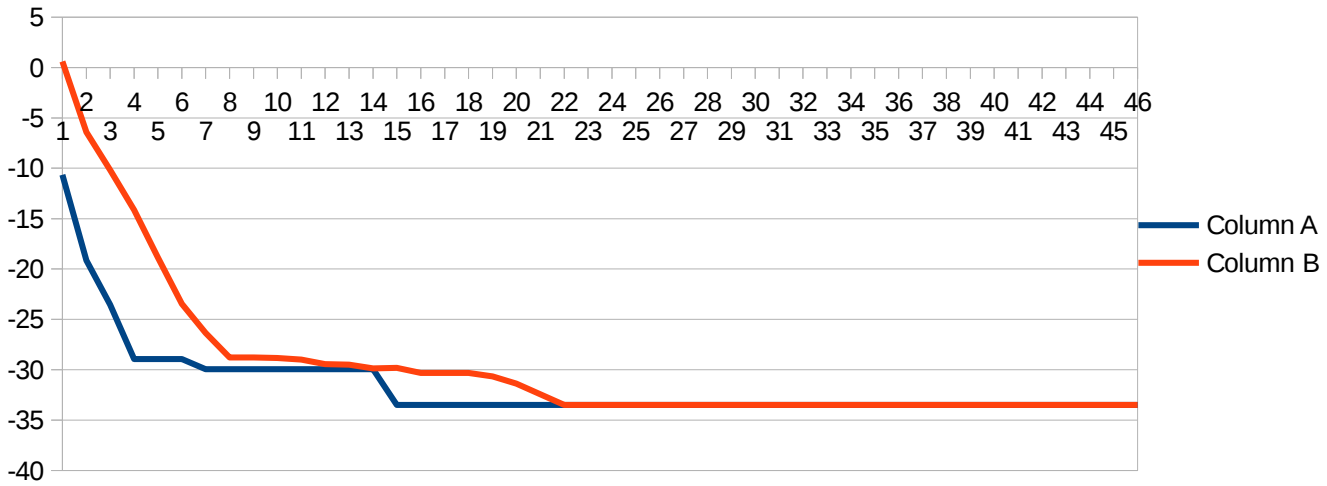
TASK1:

```
public class Main {
    public static Double[][] generation = new Double[20][20];
    public static Double[][] good = new Double[10][20];
    public static Double[][] childes = new Double[10][20];
    public static Integer childCount = 0;
    public static Integer checkCount = 0;
    public static Double prevMax = 0.0;
    public static void main(String[] args) {
        createGenerate();
        while(true) {
            Arrays.sort(generation, new Comparator<Double[]>() {
                @Override
                public int compare(Double[] o1, Double[] o2) {
                    Double int1 = fitness(o1);
                    Double int2 = fitness(o2);
                    return (int)(int1 - int2) * 100;
                }
            });
            Double max = fitness(generation[0]);
            if(prevMax.equals(max))
               checkCount ++;
            else
                checkCount = 0;
            prevMax = max;
            System.out.printf("%.3f %.3f", max,average());
            System.out.println();
            if(checkCount > 30) {
                return;
            }
            setGood();
            Random random = new Random();
            childCount = 0;
            for (int i = 0; i < 5; i++) {
                createChildes(good[random.nextInt(10)], good[random.nextInt(10)]);
            }
            createNewGeneration();
        }
    }
    public static void createGenerate() {
        Random random = new Random();
        for(int i=0;i<20;i++) {
            for (int j = 0; j < 20; j++) {
                generation[i][j] = random.nextInt(1000) / 100.0 - 5;
            }
        }
    }
    public static Double fitness(Double[] obj) {
        Double sum = 0.0;
        for (int i=0; i<obj.length;i++) {
            sum += obj[i] * Math.sin(Math.abs(obj[i]));
        }
        return sum;
    }
    public static void setGood() {
        for(int i =0; i< 10; i++) {
            good[i] = generation[i];
        }
    }
```

```
    public static void createChildes(Double[] parent1, Double[] parent2) {
            Random random = new Random();
        Integer delimiter = random.nextInt(20);
        Double[] child1 = new Double[20];
        Double[] child2 = new Double[20];
        for(int i = 0;i<20;i++) {
            if(i<=delimiter)
                child1[i] = parent1[i];
            else
                child1[i] = parent2[i];
        }
        for(int i = 0;i<20;i++) {
            if(i<=delimiter)
                child2[i] = parent2[i];
            else
                child2[i] = parent1[i];
        }
        childes[childCount] = child1;
        childCount++;
        childes[childCount] = child2;
        childCount++;
    }
    public static void createNewGeneration() {
        for (int i=0;i<20;i++) {
            if(i<10)
            generation[i] = good[i];
            else
                generation[i] = childes[i-10];
        }
    }
    public static Double average() {
        Double s = 0.0;
        for (int i=0;i<20;i++) {
            s+= fitness(generation[i]);
        }
        Double average = s / 100.0;
        return average;
    }
}
```



f

TASK2: