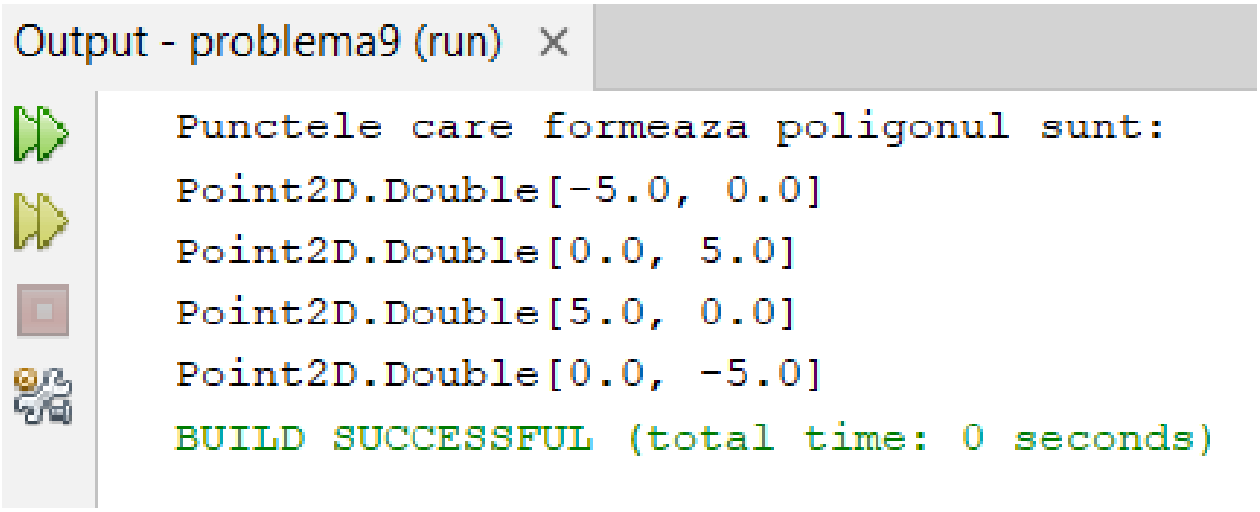
Tudor Vlăduț-Alexandru

Problema 9

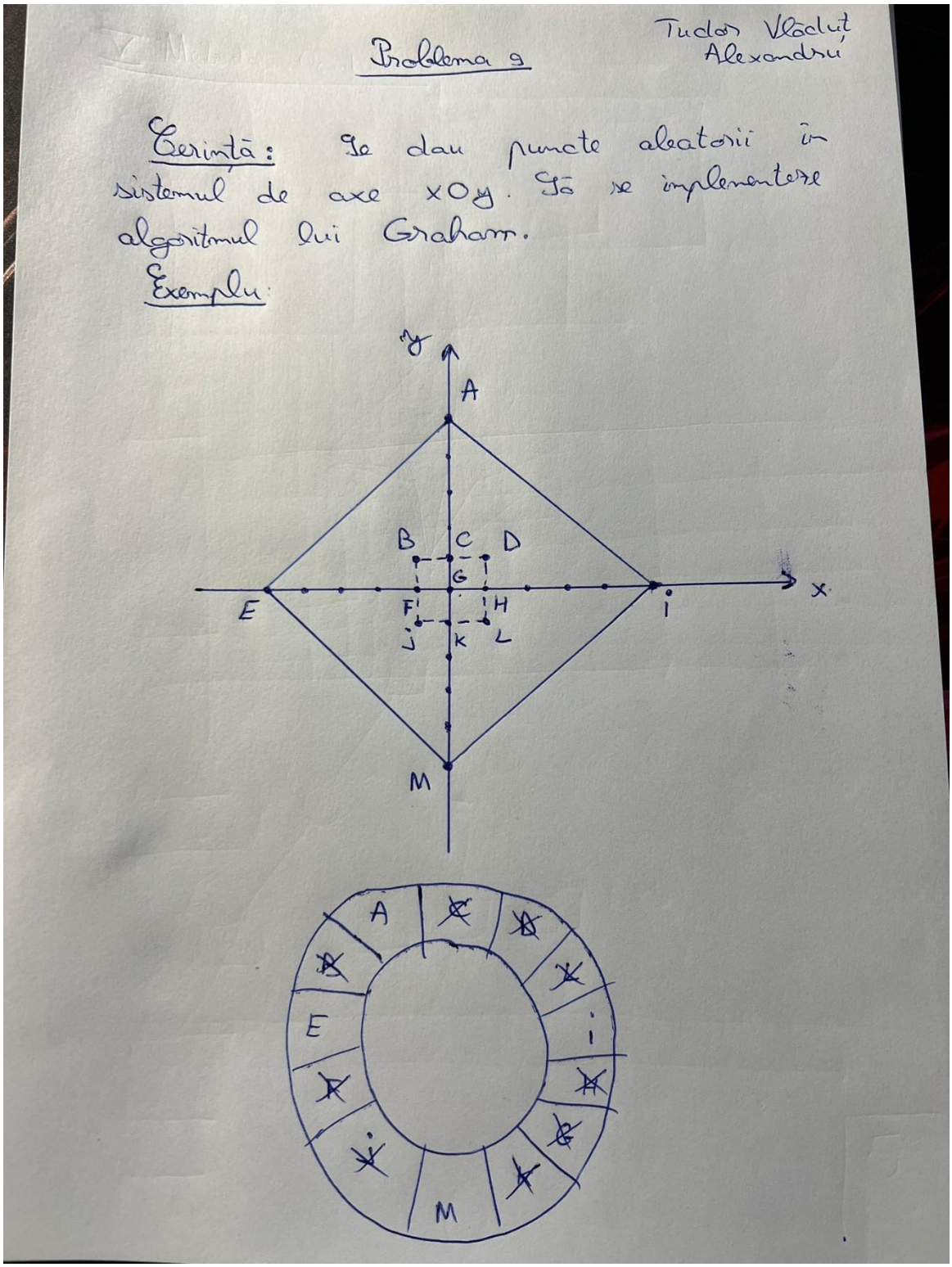
Algoritmul lui Graham

*Bonus: Țin să menționez că am fost prezent la activitatea cu numărul 3 – Atelierul Google.

Rulajul:



Pe foaie:


Tudor Vlăduț Alexandru
Problema 9
Cerință: Se dau puncte aleatorii în sistemul de axe xOy. Să se implementeze algoritmul lui Graham.
Exemplu:
y
x
A
B
C
D
E
F
G
H
I
J
K
L
M
A
E
I
M

Implementare Java:

```
package problema9;
import java.awt.geom.Point2D;
import java.util.*;
public class Problema9 {
    public static Stack<Point2D> creearePoligon(Point2D[] pts) {
    int k1, k2, N = pts.length;
    Stack<Point2D> hull = new Stack<Point2D>();
    Arrays.sort( a:pts, new PointOrder());
    Arrays.sort( a:pts,  fromIndex:1,  toIndex:N, new PolarOrder(pts[0]));
    hull.push(pts[0]);
    for (k1 = 1; k1 < N; k1++) {
      if (!pts[0].equals(pts[k1])) {
        break;
      }
    }
    if (k1 == N)
        return null;
    for (k2 = k1 + 1; k2 < N; k2++) {
      if (collinear(pts[0], pts[k1], pts[k2]) != 0) {
        break;
      }
    }
    hull.push(pts[k2 - 1]);
    for (int i = k2; i < N; i++) {
      Point2D top = hull.pop();
      while (collinear( a:hull.peek(),  b:top, pts[i]) <= 0) {
        top = hull.pop();
      }
      hull.push( item:top);
      hull.push(pts[i]);
    }
    return hull;
  }
```

```
private static class PolarOrder implements Comparator<Point2D> {
  Point2D pt;
  public PolarOrder(Point2D pt) {
    this.pt = pt;
  }
  @Override
  public int compare(Point2D q1, Point2D q2) {
    double dx1 = q1.getX() - pt.getX(), dy1 = q1.getY() - pt.getY();
    double dx2 = q2.getX() - pt.getX(), dy2 = q2.getY() - pt.getY();
    if (dy1 >= 0 && dy2 < 0)
        return -1;
    else if (dy2 >= 0 && dy1 < 0)
        return +1;
    else if (dy1 == 0 && dy2 == 0) {
      if (dx1 >= 0 && dx2 < 0)
          return -1;
      else if (dx2 >= 0 && dx1 < 0)
          return +1;
      else
          return 0;
    }
    else return -collinear( a:pt,  b:q1,  c:q2);
  }
}
```

```
private static class PointOrder implements Comparator<Point2D> {
  @Override
  public int compare(Point2D q1, Point2D q2) {
    if (q1.getY() < q2.getY())
        return -1;
    if (q1.getY() == q2.getY()) {
      if (q1.getX() < q2.getX())
          return -1;
      else if (q1.getX() > q2.getX())
          return 1;
      else
          return 0;
    }
    return 1;
  }
}
private static int collinear(Point2D a, Point2D b, Point2D c) {
  double area = (b.getX() - a.getX()) * (c.getY() - a.getY()) - (b.getY() - a.getY()) * (c.getX() - a.getX());
  return (int) Math.signum( d:area);
}
```

```
  public static void main(String[] args) {

    Point2D[] pts = new Point2D[13];

    pts[0] = new Point2D.Double( x: 0,  y: 5);
    pts[1] = new Point2D.Double( x: -1,  y: 1);
    pts[2] = new Point2D.Double( x: 0,  y: 1);
    pts[3] = new Point2D.Double( x: 1,  y: 1);
    pts[4] = new Point2D.Double( x: -5,  y: 0);
    pts[5] = new Point2D.Double( x: -1,  y: 0);
    pts[6] = new Point2D.Double( x: 0,  y: 0);
    pts[7] = new Point2D.Double( x: 1,  y: 0);
    pts[8] = new Point2D.Double( x: 5,  y: 0);
    pts[9] = new Point2D.Double( x: -1,  y: -1);
    pts[10] = new Point2D.Double( x: 0,  y: -1);
    pts[11] = new Point2D.Double( x: 1,  y: -1);
    pts[12] = new Point2D.Double( x: 0,  y: -5);

    Stack<Point2D> hull = crearePoligon(pts);
    System.out.println( x: "Punctele care formeaza poligonul sunt:");
    while (!hull.isEmpty())
        System.out.println( x: hull.pop());
  }
}
```