Algoritmos y Estructura de Datos III

Una introducción a Java - Ejemplo

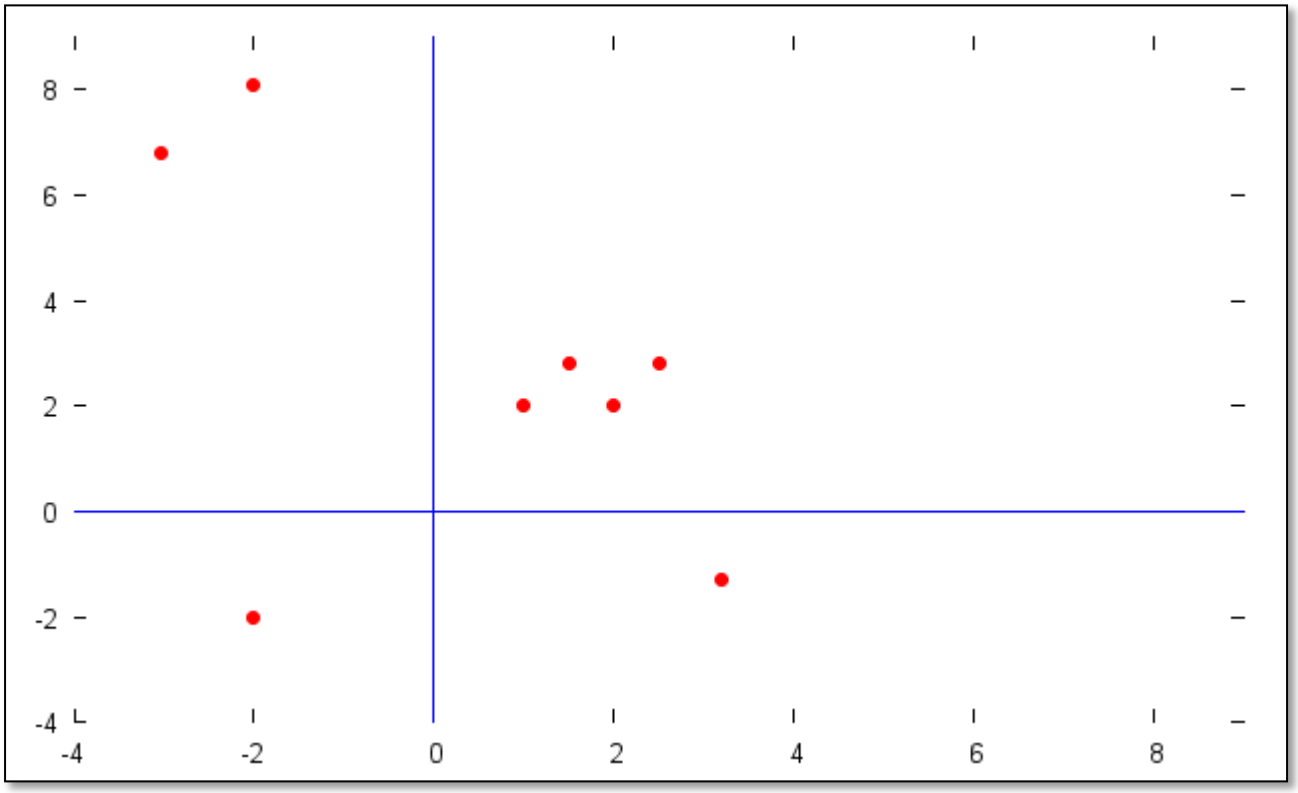
Prof. Cristian Cappo

Problema

- Dado N puntos de plano bidimensional, encontrar la distancia mínima que existe entre ellos, indicando los puntos a que se refiere esa distancia. Si hay puntos que están a esa distancia mínima, imprimir todos.
- Considerar que los puntos pueden tener coordenadas positivas o negativas.
- Los datos son leídos desde un archivo, el cual contiene la coordenada de cada punto para el eje X y el eje Y, separados por uno o más espacios.

Ejemplo:

1 2
1.5 2.8
3.2 -1.3
-3.03 6.78
-2.01 8.08
-2 -2
2 2
2.5 2.8



¿Dónde empezamos?

- La distancia que utilizamos es la distancia Euclidiana simple.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- ¿Cómo organizar los datos?
  - ¿Cómo diseño los puntos?

    En C: puedo hacer que sea una estructura.

    ```
    typedef  struct { float x; float y; }  TPunto;
    ```

  - ¿Qué es el plano?

    Podemos considerar un arreglo de N datos de tipo ***TPunto***

    ```
    TPunto * Plano = (TPunto *) malloc (N *sizeof(TPunto));
    ```

Encontrar el mínimo

- Encontrar todas las distancias posibles y a la vez encontrar la distancia mínima.
- Volver a generar todas las distancias posibles e imprimir aquellas que corresponden al mínimo encontrado anteriormente.

main

```
int  main ()
{
    int N, c ;
    float x, y;

    scanf("%d", &N);

    TPunto * Plano = (TPunto *) malloc ( N * sizeof(TPunto) );

    if ( Plano == NULL ) exit(1);

    c = 0;
    while ( c < N ) {
       scanf("%f%f",&x,&y);
       Plano[c].x = x; Plano[c].y = y;
       c++;
    }

    distanciaMinima(Plano, N);

    return 0;
}
```

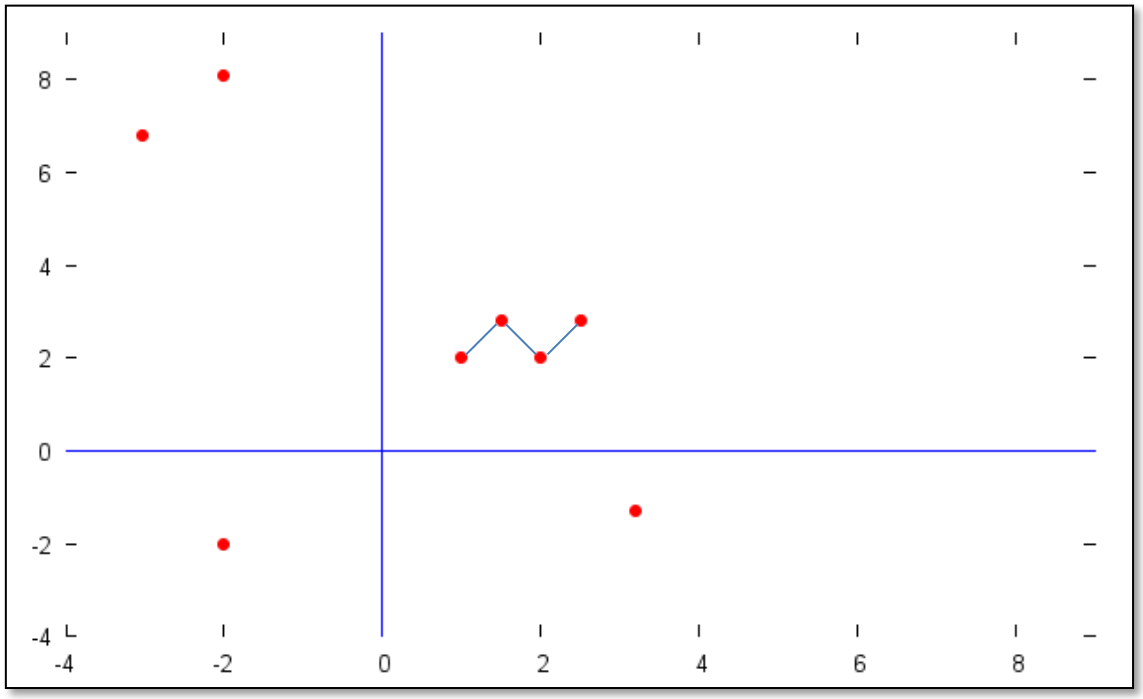
El proceso en sí

```
void  distanciaMinima ( TPunto * Plano, int N )
{
    int i, j;
    float min = calcular_distancia(Plano[0], Plano[1]) , dist ;

    for ( i = 0 ; i < N ; i++ )
        for ( j=i+1; j < N; j++ ) {
            dist = calcular_distancia( Plano[i], Plano[j] );
            if  ( d < min ) min = d;
        }

    printf("Distancia minima %f entre los puntos:\n", min);
    for ( i = 0 ; i < N ; i++ )
        for ( j=i+1; j < N ; j++ ) {
            dist = calcular_distancia( Plano[i], Plano[j] );
            if ( d == min )
               printf("\t[%f,%f ]-[%f,%f ]",..);
        }
}
```

Funcionamiento

Entrada

```
8
1 2
1.5 2.8
3.2 -1.3
-3.03 6.78
-2.01 8.08
-2 -2
2 2
2.5 2.8
```



Salida

```
Distancia Minima 0.943398 entre los puntos:
        [1.00,2.00]--[1.50,2.80]
        [1.50,2.80]--[2.00,2.00]
        [2.00,2.00]--[2.50,2.80]
```

Ahora en Java (1) ☺

```
class Punto {
   float x;
   float y;

   Punto ( float x, float y) {
       this.x = x;
        this.y = y;
   }
   static float calcular_distancia( Punto pt1, Punto pt2 ) {
     return (float) Math.sqrt( ((pt1.x - pt2.x) * (pt1.x -  pt2.x)) +
   ((pt1.y - pt2.y) * (pt1.y - pt2.y)) );
   }
}
```

Ahora en Java (2)

```
import java.util.*;

public class MinPlano {

    public static void main ( String [] args ) {
        int N;
        int c;
        float x, y;
        Scanner s = new Scanner(System.in);
        s.useLocale(Locale.US); //Para considerar el punto(.) como punto decimal

        N = s.nextInt();
        Punto [] Plano = new Punto[N];
        c = 0;
        while ( c < N  && s.hasNextLine() ) {
            x = s.nextFloat();
            y = s.nextFloat();
            Plano[c] = new Punto(x,y);
            c++;
        }
        distanciaMinima(Plano);
    }

.. // continua
```

Ahora en Java (3)

```
 public static void distanciaMinima( Punto [] Plano ) {
   int i, j;
   float min = Punto.calcular_distancia(Plano[0],Plano[1]), dist;
   for ( i = 0 ; i < Plano.length ; i++ )
      for ( j=i+1; j < Plano.length; j++ ) {
          dist = Punto.calcular_distancia( Plano[i], Plano[j] );
          if  ( dist < min ) min = dist;
      }
   System.out.printf("Distancia minima %f entre los puntos:\n", min);

   for ( i = 0 ; i < Plano.length ; i++ )
       for ( j=i+1; j < Plano.length ; j++ ) {
          dist = Punto.calcular_distancia( Plano[i], Plano[j] );
          if ( dist == min )
            System.out.printf("\t[%.2f,%.2f]--  [%.2f,%.2f]\n", ..);
      }
 }
} // Fin  de la clase MinPlano
```

Distribución en el archivo MinJava.java