Question 3 of 3

0:02:52

DESC

RULES

README

SETTINGS

Codewriting

You have an array `p` of points on a Cartesian plane. Find and return the minimum possible Euclidian distance between two points with different indices in `p`.

Example

For `p = [[0, 11], [-7, 1], [-5, -3]]`, the output should be

`solution(p) = 4.472135955`.

Input/Output

- **[execution time limit] 3 seconds (java)**
- **[input] array.array.integer p**

  Every inner array `p[i]` contains exactly 2 integers: the `x` and `y` coordinates of the `ith` point.

  Guaranteed constraints:

  `2 ≤ p.length ≤ 2 · 10`^{4},

  `p[i].length = 2`,

  `|p[i][j]| ≤ 10`^{7}.

- **[output] float**

  The minimum possible distance between two points with different indices in `p`.

  Your answer will be considered correct if its absolute error doesn't exceed `10`$^{-5}$.

[Java] Syntax Tips

```
// Prints help message to the console
// Returns a string
//
// Globals declared here will cause a compilation e
// declare variables inside the function instead!
String helloworld(String name) {
    System.out.println("This prints to the console
    return "Hello, " + name;
}
```

main.java

Java

```
static double find(double x, double y,
                   int [][] p)
{
    double mind = 0;

    for(int i = 0; i < p.length; i++)
    {
        double a = p[i][0], b = p[i][1];
        mind += Math.sqrt((x - a) * (x - a) +
                          (y - b) * (y - b));
    }
    return mind;
}

double solution(int[][] p) {

    // Calculate the centroid
    double x = 0, y = 0;
    for(int i = 0; i < p.length; i++)
    {
        x += p[i][0];
        y += p[i][1];
    }
```

TESTS CUSTOM TESTS RUN TESTS

Test 1

MORE

0/200 VIEW DIFF SUBMIT REVIEW YOUR ANSWERS