Convex Hull ★

Points: 553.1800000000001 Rank: 1150

Problem Submissions Leaderboard

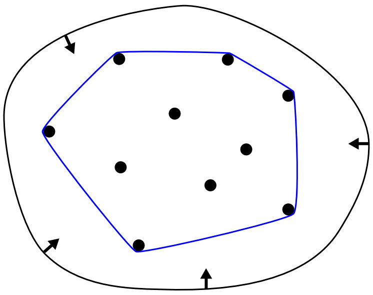
RATE THIS CHALLENGE

★★★★★

Contributed by Abhiranjan Kumar

Convex Hull of a set of points, in 2D plane, is a convex polygon with minimum area such that each point lies either on the boundary of polygon or inside it.

Let's consider a 2D plane, where we plug pegs at the points mentioned. We enclose all the pegs with a elastic band and then release it to take its shape. The closed structure formed by elastic band is similar to that of convex hull.



In the above figure, convex hull of the points, represented as dots, is the polygon formed by blue line.

Tasks

Given a set of N points, Find the perimeter of the convex hull for the points.

Input Format

First line of input will contain a integer, N, number of points. Then follow N lines where each line contains the coordinate, x_i y_i, of i^{th} point.

Output Format

Print the perimeter of convex hull for the given set of points. An error margin of +/- 0.2 is acceptable.

Constraints

$3 <= N <= 10^4$

$0 <= x_i, y_i <= 10^4$

There exists, at least, three points which are non-colinear.

Sample Input

```
6
1 1
2 5
3 3
5 3
3 2
2 2
```

Sample Output

```
12.2
```

Explanation

For the given set of points in sample input, the convex hull is formed by the triangle whose vertices are given by `(1, 1), (2, 5), (5, 3)`. Here perimeter of the hull is 12.200792856.

Change Theme Language Haskell

```
import Control.Monad
import Data.List (sortBy, foldl')
import Data.Ord (comparing)

dotProduct :: (Double, Double) -> (Double, Double) -> Double
dotProduct (x1, y1) (x2, y2) = x1*x2 + y1*y2


crossProduct :: (Double, Double) -> (Double, Double) -> Double
crossProduct (x1, y1) (x2, y2) = x1*y2 - y1*x2

getAngle :: (Double, Double) -> (Double, Double) -> Double
getAngle a b = let d = dotProduct a b
                   c = crossProduct a b
                   angle = atan2 c d
                in if angle < 0
                      then angle + 2 * pi
                      else angle


sortByAngle :: [(Double, Double)] -> [(Double, Double)]
sortByAngle []  = []
sortByAngle [a] = [a]
sortByAngle as  = sortBy f as
    where f xy1 xy2 = let angle1 = getAngle (1,0) xy1
                          angle2 = getAngle (1,0) xy2
                       in if angle1 == angle2
                             then compare (l xy1)  (l xy2)
```

Line: 66 Col: 1

Upload Code as File □ Test against custom input Run Code Submit Code

Contest Calendar | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature