Md. Asad Chowdhury Dipu

Id: 2019-1-60-093

Section: 1

Ans:

#include <iostream>

#include <stack>

#include<stdio.h>

#include <stdlib.h>

using namespace std;

struct Point

{

int x, y;

};

Point p0;

Point Next\_Top(stack<Point> &S)

{

Point p = S.top();

S.pop();

Point res = S.top();

S.push(p);

return res;

}

void swap(Point &p1, Point &p2)

{

Point temp = p1;

p1 = p2;

p2 = temp;

}

int Dis\_of\_pints(Point p1, Point p2)

{

int val=(p1.x - p2.x)\*(p1.x - p2.x) + (p1.y - p2.y)\*(p1.y - p2.y);

return val;

}

int Orientation(Point p, Point q, Point r)

{

int val = (q.y - p.y) \* (r.x - q.x) -

(q.x - p.x) \* (r.y - q.y);

if (val == 0)

return 0;

if (val > 0)

{

return 1;

}

else

{

return 2;

}

}

int compare(const void \*vp1, const void \*vp2)

{

Point \*p1 = (Point \*)vp1;

Point \*p2 = (Point \*)vp2;

int return\_orientation = Orientation(p0, \*p1, \*p2);

if (return\_orientation == 0)

return (Dis\_of\_pints(p0, \*p2) >= Dis\_of\_pints(p0, \*p1))? -1 : 1;

return (return\_orientation == 2)? -1: 1;

}

void Convex\_Hull(Point points[], int n)

{

int Min\_Y = points[0].y, min = 0;

for (int i = 1; i < n; i++)

{

int y = points[i].y;

if ((y < Min\_Y) || (Min\_Y == y &&

points[i].x < points[min].x))

Min\_Y = points[i].y, min = i;

}

swap(points[0], points[min]);

p0 = points[0];

qsort(&points[1], n-1, sizeof(Point), compare);

int m = 1;

for (int i=1; i<n; i++)

{

while (i < n-1 && Orientation(p0, points[i], points[i+1]) == 0)

i++;

points[m] = points[i];

m++;

}

if (m < 3)

return;

stack<Point> S;

S.push(points[0]);

S.push(points[1]);

S.push(points[2]);

for (int i = 3; i < m; i++)

{

while (S.size()>1 && Orientation(Next\_Top(S), S.top(), points[i]) != 2)

S.pop();

S.push(points[i]);

}

while (!S.empty())

{

Point p = S.top();

printf("(%d,%d)\n",p.x,p.y);

S.pop();

}

}

int main()

{

int i,n;

printf("Number of Points: ");

scanf("%d",&n);

Point points[n];

printf("Enter points X & Y: \n");

for(i=0; i<n; i++)

{

printf("Pair %d: ", i+1);

scanf("%d%d", &points[i].x, &points[i].y);

}

Convex\_Hull(points, n);

return 0;

}