Звіт по лабораторній роботі №1

З предмету Комп`ютерна графіка

Студента групи ІПС-31

Олійника Микита

**Тема**: приналежність точки опуклому многокутнику

**Алгоритм**:

1. Знайдемо найлівішу вершину даного многокутника
2. Відсортуємо всі вершини многокутника за полярним кутом відносно знайденої
3. Для кожного запиту на перевірку на приналежність точки многокутника:
   1. Знайдемо бінарним пошуком вершину многокутника у секторі якої лежить запит
   2. Перевіримо по яку сторону від ребра многокутника, що виходить зі знайденої вершини, лежить запит

**Код**:

#include <iostream>  
#include <fstream>  
#include <vector>  
#include <algorithm>  
  
using namespace std;  
  
struct Point  
{  
 int x, y;  
 Point(int a, int b) : x(a), y(b) {}  
 bool operator<(Point p)  
 {  
 return x < p.x || (x == p.x && y < p.y); // для пошуку найлівішої точки  
 }  
};  
  
int area(Point a, Point b, Point c) // орієнтована площа трикутника  
{  
 return (b.x - a.x)\*(c.y - a.y) - (b.y - a.y)\*(c.x - a.x);  
}  
  
bool right(Point a, Point b, Point c) // предикат «за годинниковою стрілкою»  
{  
 return area(a, b, c) <= 0;  
}  
  
struct Sorter  
{  
 Point start;  
 Sorter(Point u) : start(u) {}  
 bool operator()(Point u, Point v)  
 {  
 return area(start, u, v) <= 0; // для сортування за полярним кутом відносно заданої точки  
 }  
};  
  
int main()  
{  
 ifstream fin("../input.txt");  
  
 int n, start = 0;  
 fin >> n;  
 vector < Point > polygon;  
  
 for (int i = 0; i < n; ++i)  
 {  
 int x, y;  
 fin >> x >> y;  
 Point p(x, y);  
 polygon.emplace\_back(p);  
 if (p < polygon[start]) // шукаємо найлівішу точку  
 start = i;  
 }  
  
 swap(polygon[start], polygon[0]);  
 sort(polygon.begin()+1, polygon.end(), Sorter(polygon[0])); // сортуємо точки   
  
 while (true)  
 {  
 cout << "X, Y:" << endl;  
 int x, y;  
 cin >> x >> y;  
 Point input(x, y);  
  
 int l = 0, r = n;  
 while (r - l > 1) // бінарний пошук сектора якому належить запит  
 {  
 int m = (l + r) >> 1;  
 if (right(polygon[0], polygon[m], input))  
 l = m;  
 else  
 r = m;  
 }  
 if (!right(polygon[0], polygon[1], input) || !right(polygon[l], polygon[(l + 1) % n], input)) // перевірка на приналежність многокутнику  
 cout << "OUT\n";  
 else  
 cout << "IN\n";  
 }  
 return 0;  
}