МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
Кафедра програмної інженерії та інформаційних технологій управління

Звіт з лабораторної роботи № 7
з дисципліни «Основи теорії алгоритмів»

Виконав:
ст. гр. КН-221в
Шулюпов Є.Р.
Перевірила:
доцент каф. ПІТУ
Солонська С.В

Харків
2022

ТЕМА: ГЕОМЕТРИЧНІ АЛГОРИТМИ

ЗАВДАННЯ НА ЛАБОРАТОРНУ РОБОТУ

Розробити програму, яка читає з клавіатури число N ($1 < N < 256$) та N пар дійсних чисел — координати точок на площині. Програма виконує один за алгоритмів згідно варіанту.

Побудувати опуклу оболонку наданих точок алгоритмом Джарвіса.

МЕТА РОБОТИ

Ознайомлення із основними геометричними алгоритмами.

1 ОСНОВНІ ТЕОРЕТИЧНІ ПОЛОЖЕННЯ

Важливу роль в багатьох геометричних алгоритмах відіграє визначення напрямку повороту одного вектора до другого. Для розв’язання даної задачі часто використовують поняття векторного (чи, як його називають ще, псевдоскалярного) добутку.

Алгоритм Джарвіса [1].

Побудова опуклої оболонки даної множини починається з точки, яка гарантовано буде входити до опуклої оболонки. Очевидно, що сама ліва нижня точка підходить під цю умову, позначимо її [1].

Далі знаходиться наступна точка (де – крок алгоритму) у випадку обходу проти годинникової стрілки. Вона, очевидно, володіє властивістю, що інші точки лежать зліва від вектора [1].

Пошук точки, що включається на -м кроці алгоритму можна здійснити так. Вибираємо будь-яку точку не з оболонки як претендента на включення, позначимо її . Потім послідовно порівнюємо точку з точками не з оболонки (а також з початковою точкою оболонки , так як нам необхідно в кінці побудови оболонки «зімкнути» її, тобто із поточної точки знову прийти в точку). Для порівняння точок будемо використовувати

знак векторного добутку . Якщо для одної із точок, що розглядаються, вказаний вираз менший від нуля, то вважаємо її претендентом на включення і продовжуємо перевірку інших точок. Якщо ж значення векторного добутку рівне нулю, то вибираємо ту точку, яка розташована далі від точки [2].

Алгоритм обходу Грехема був описаний ним в одній із його перших робіт та присвяченій питанню розробки ефективних геометричних алгоритмів.

Алгоритм починає свою роботу з находження початкової точки із заданої множини точок (не меншої трьох), яка гарантовано є вершиною опуклої оболонки. В якості такої береться сама ліва із самих нижніх точок множини [1].

На етапі попереднього сортування всі точки вихідної множини (окрім) сортуються за зростанням полярного кута, утвореного кожною поточною точкою , відносно точки . Якщо дві точки мають однаковий полярний кут, то залишаємо для подальшого розгляду ту точку, відстань від якої до точки більша, так як точка з меншою відстанню до заздалегідь не потрапляє до оболонки. Відсортовані точки поміщаються в стек. Відмітимо, що таке сортування можна здійснити за допомогою векторного добутку.

На етапі побудови опуклої оболонки алгоритм виконує покрокову обробку відсортованих точок, формуючи оболонку шляхом видалення тих точок, в яких не здійснюючи лівий поворот. Це очевидно, так як оболонка будується проти годинникової стрілки, починаючи від , значить, рух здійснюється завжди наліво [1].

На кожному кроці даного етапу перевіряється взаємне розташування трьох точок (спочатку вибираються в перші дві точки у відсортованому списку). Перевірка полягає у визначенні , чи утворюють точки, які перевіряються, лівий або правий поворот. Таку перевірку також можна здійснити за допомогою векторного добутку [2].

2 ОПИСАННЯ РОЗРОБЛЕНОГО ЗАСТОСУНКУ

Програма реалізована в одному файлі source.cpp

```
#include<iostream>
#include<vector>
using namespace std;
```

```
using namespace std;
//Структура точки
struct Point {
    int x, y;
};
//Обчислення позиції точки
int cal_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; //колінеарні
        return (val > 0) ? 1 : 2; //за або проти годинниковою стрілкою
}

void convexHull(Point points[], int n) {
    if (n < 3) return;
    vector<Point> hull;
    //обчислення лівої точки
    int l = 0;
    for (int i = 1; i < n; i++)
    {
        if (points[i].x < points[l].x)
            l = i;
    }

    //рухаємося за годинниковою стрілкою
    int p = l, q;
    do {
        //додавання поточної точки до результату
        hull.push_back(points[p]);
        q = (p + 1) % n;
        for (int i = 0; i < n; i++) {
            if (cal_orientation(points[p], points[i], points[q]) == 2)
                q = i;
        }
        p = q;
    } while (p != l); //якщо не дійшов до першої точки
    for (int i = 0; i < hull.size(); i++)
        cout << "(" << hull[i].x << ", " << hull[i].y << ")\n";
}
int main()
{
    setlocale(LC_ALL, "Ukr");
    cout << "Введіть значення N, стільки пар координат введіть" << endl;
    Point points[256];
    int n;
    cin >> n;
    for (int i = 0; i < n; i++)
    {
        int x, y;
        cout << "Введіть координату x:" << endl;
        cin >> x;
        points[i].x = x;
        cout << "Введіть координату y:" << endl;
        cin >> y;
        points[i].y = y;
    }
    cout << "Опукла оболонка складається з точок:" << endl;
    convexHull(points, n);
    return 0;
       }
```

Результат роботи представлений на рисунку 2.1.

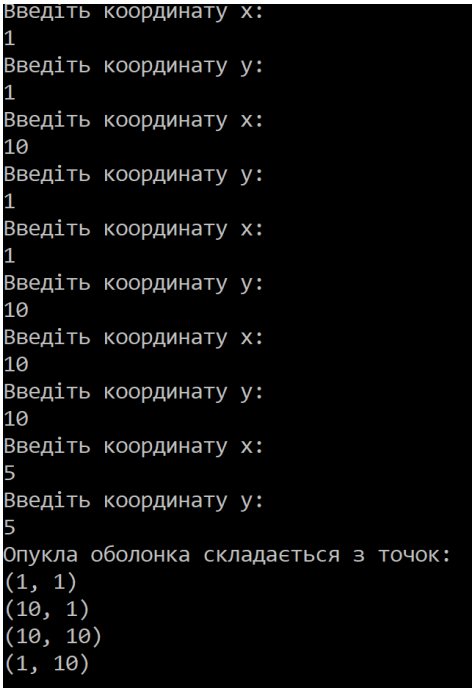

Рисунок 2.1 – Результат

ВИСНОВКИ

Під час цієї лабораторної роботи була розроблена програма, яка виконує геометричний алгоритм згідно варіанту завдання, а саме будування опуклої оболонки на площині із точок, із виведення на консольний застосунок тих точок, що утворюють дану оболонку. Дана програма виконує всі необхідні функції, які описані у завданні до даної лабораторної роботи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 Методичні вказівки до виконання лабораторних робіт з курсу "Алгоритми і структури даних": для студентів, які навчаються за спец. 121 "Інженерія програмного забезпечення" [Електронний ресурс] / уклад. Н. К. Стратієнко, І. О. Бородіна ; Харківський політехнічний інститут, національний технічний університет університет –

Електрон. текстові дані. – Харків, 2017. – 36 с. – Режим доступу: http://repository.kpi.kharkov.ua/handle/KhPI-Press/26426.

2. Алгоритми і структури даних: практикум: навч. посіб./ Н.К. Стратієнко, М.Д. Годлевський, І.О. Бородіна.- Харьков: НТУ"ХПИ", 2017. - 224 с.