Университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №2

по «Алгоритмам и структурам данных»

Выполнил:

Студент группы P3210

Кадыров А.Х.

Преподаватели:

Косяков М.С.

Тараканов Д.С.

Санкт-Петербург

2019

Задача №1207 «Медиана на плоскости»

Пояснение к примененному алгоритму:

Найдем самую крайнюю точку и узнаем угол до каждой из оставшихся точек и отсортируем эти значения по величине угла. В таком случае мы сможем найти ту точку через которую можно провести прямую, чтобы разделить плоскость так, чтобы половина точек находились выше этой прямой и точки, а вторая половина - ниже.

Код:

```
#include <iostream>
#include <vector>

using namespace std;

struct coordinate {
  int x;
  int y;
  int number;
  double k;
};

double koef(coordinate p, int x, int y) {
  return (p.y - y) / (double)(p.x - x);
}

vector<coordinate> merge(vector<coordinate> points, int l1, int r1, int l2,
                         int r2) {
  vector<coordinate> pointsSorted = points;

  int i1 = l1;
  int i2 = l2;
  int ind = l1;

  for (int i = 0; i < r1 - l1 + 1 + r2 - l2 + 1; ++i) {
```

```
    if (i1 <= r1 && (i2 > r2 || points[i1].k < points[i2].k)) {
      pointsSorted[ind++] = points[i1++];
    } else {
      pointsSorted[ind++] = points[i2++];
    }
  }
  return pointsSorted;
}

void merge_sort(vector<coordinate> &points, int left, int right) {
  if (right - left + 1 < 2) {
    return;
  }

  int middle = (left + right) / 2;
  merge_sort(points, left, middle);
  merge_sort(points, middle + 1, right);
  points = merge(points, left, middle, middle + 1, right);
}

int main() {
  int n;
  int x;
  int y;
  int minX = INT_MAX;
  int minY = INT_MAX;
  int minI;
  cin >> n;
  vector<coordinate> points(n);
  for (int i = 0; i < n; i++) {
    cin >> x >> y;
    points[i].x = x;
    points[i].y = y;
```

```
    points[i].number = i;
    if ((minX == points[i].x && minY > points[i].y) || minX > points[i].x) {
      minX = points[i].x;
      minY = points[i].y;
      minI = i;
    }
  }
  for (int i = 0; i < n; i++) {
    if (i == minI) {
      points[i].k = INT_MIN;
      continue;
    }
    if (minX == points[i].x) {
      points[i].k = INT_MAX;
      continue;
    }
    points[i].k = koef(points[i], minX, minY);
  }
  merge_sort(points, 0, n - 1);

  cout << points[0].number + 1 << " " << points[n / 2].number + 1 << endl;

  return 0;
}
```