**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчет

**по лабораторной работе №2**

**по дисциплине «Алгоритмы и структуры данных»**

тема: Реализация метода «Разделяй и властвуй»

|  |  |  |
| --- | --- | --- |
| Студентка гр. 1302 |  | Королев П.С. |
| Преподаватель |  | Родионова Е.А. |

**Санкт-Петербург**

**2022**

**Цель работы**

Решить задачу с использованием принципа «Разделяй и властвуй».

Определить теоретическую асимптотическую сложность решения.

Эмпирически оценить временную сложность решения для «average-case».

**Основные теоретические положения**

В рекурсивную функцию подается множество точек, функция разбивает это множество пополам и вызывается рекурсивно от каждой из половин, затем выполняется слияние двух половин и проверка на наличие результата, состоящего из точек, находящихся в разных половинах.

Асимптотическая сложность – О(nlogn).

**Результатработы программы**

**

**Листинг**

#include <iostream>

#include <algorithm>

#include <cmath>

#include <conio.h>

#include <time.h>

using namespace std;

double MINdistance = 100000000;

int X1 = 0, X2 = 0, Y1 = 0, Y2 = 0;

const int numberOFsnails = 15;

class snail {

public:

int x, y;

};

bool sortingX(snail a, snail b) // X sorting conditions

{

return a.x < b.x || (a.x == b.x && a.y < b.y);

}

bool sortingY(const snail& a, const snail& b) // Y sorting conditions

{

return a.y < b.y;

}

double distance(snail a, snail b) // Getting the distation

{

return sqrt(pow((a.x - b.x), 2) + pow((a.y - b.y), 2));

}

void new\_answer(snail a, snail b) // Сomparison the answers

{

double distation1 = distance(a, b);

if (distation1 < MINdistance)

{

MINdistance = distation1;

X1 = a.x;

Y1 = a.y;

X2 = b.x;

Y2 = b.y;

}

}

void recursion(int left, int right, snail\* a) // Sorting and recursive checking

{

if (right - left <= 2)

{

for (int i = left; i <= right; i++)

for (int j = i + 1; j <= right; j++)

{

new\_answer(a[i], a[j]);

}

sort(a + left, a + right + 1, &sortingY);

return;

}

int mid = (left + right) / 2;

int midx = a[mid].x;

recursion(left, mid, a);

recursion(mid + 1, right, a);

snail\* buff = new snail[numberOFsnails];

merge(a + left, a + mid + 1, a + mid + 1, a + right + 1, buff, &sortingY);

copy(buff, buff + right - left + 1, a + left);

int counter = 0;

for (int i = left; i <= right; i++)

if (abs(a[i].x - midx) < MINdistance)

{

for (int j = counter - 1; j >= 0 && (a[i].y - MINdistance < buff[j].y); j--)

new\_answer(a[i], buff[j]);

buff[counter++] = a[i];

}

}

int main() {

srand(time(NULL));

snail a[numberOFsnails];

for (int i = 0; i < numberOFsnails; i++)

{

a[i].x = rand() % 150;

a[i].y = rand() % 150;

}

sort(a, a + size(a), &sortingX);

recursion(0, size(a) - 1, a);

double time = (MINdistance / 2) / 0.01;

cout << "They will meet in " << time << " seconds" << endl;

}

**Вывод**

В ходе данной работы я узнал много нового о принципе «разделяй и властвуй», а так же начал изучать рекурсивные функции на практике.