Министерство образования и науки Российской Федерации

Московский государственный технический университет имени Н.Э. Баумана

Факультет «Робототехника и комлексная автоматизация»

Кафедра «Системы автоматизированного проектирования»

**Отчет по лабораторной работе**

По курсу «Объектно-ориентированное программирование»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Выполнил: |  |  | Студент | Петраков С.А. |
|  |  |  | Группа | РК6-26Б |
|  |  |  |  |  |
| Проверил: |  |  |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |  | Дата | \_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |  | Подпись | \_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |  |  |  |

Москва, 2020 г.

**Вариант ПЯВУ Р16**

**Задание:**

Разработать объектно-ориентированную программу поиска в любом заданном наборе точек плоскости пары наиболее отдаленных точек, которые находятся по разные стороны от оси Х. Декартовые координаты всех точек заданного набора должны быть указаны парами целых чисел в аргументах командной строки вызова программы. Координаты полученной пары точек и расстояние между ними должны отображаться строкой потока стандартного вывода. В программе должен быть реализован класс точки с приватными полями для ее декартовых координат, публичными методами доступа к ним и конструктором инициализации их значений. Вычисление расстояния между парами точек должна обеспечивать дружественная функция этого класса. Кроме того, в программе должно быть предусмотрено динамическое распределение памяти для всех точек заданного набора и массивов их адресов.

**Текст программы:**

#include <stdio.h>

#include <math.h>

class Point

{

private:

double \_x;

double \_y;

public:

double getx()

{

return \_x;

}

double gety()

{

return \_y;

}

Point(double x, double y) :\_x(x), \_y(y) {}

friend double distance(Point\*, Point\*);

};

double distance(Point\* one, Point\* two)

{

double temp = pow(one->getx() - two->getx(), 2) + pow(one->gety() - two->gety(), 2);

return sqrt(temp);

}

int main(int argc, char\* argv[])

{

if (argc < 2)

return(-1);

int x, y;

int i, j = 0;

Point\*\* mass;

mass = new Point \* [argc];

for (i = 1; i < argc; i++)

{

sscanf(argv[i], "(%d%\*c%d)", &x, &y);

mass[j] = new Point(x, y);

j++;

}

mass[j] = NULL;

double maxl = -1, maxt;

int max1, max2;

for (i = 0; mass[i] != NULL; i++)

for (j = i + 1; mass[j] != NULL; j++)

if ((mass[i]->gety() \* mass[j]->gety()) < 0)

if (maxl < (maxt = distance(mass[i], mass[j])))

{

maxl = maxt;

max1 = i;

max2 = j;

}

if (maxl == -1)

printf("No point.\n");

else

printf("P((%lf;%lf);(%lf;%lf))=%lf.\n", mass[max1]->getx(), mass[max1]->gety(), mass[max2]->getx(), mass[max2]->gety(), maxl);

for (j = 0; mass[j] != NULL; j++)

delete mass[j];

delete[]mass;

return 0;

}

**Входные/выходные данные:**

На вход координаты точек. Выход: Минимальное расстояние между точками или не существование таких точек.

**Тесты:**

|  |  |  |
| --- | --- | --- |
| Номер | Вход | Выход |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |