Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Кафедра прикладной математики и программирования

ОТЧЕТ

о выполнении лабораторной работы № 2 по дисциплине
«Математические основы компьютерной графики»

Автор работы,
студент группы ЕТ-211
_____ Савонин М.В.
«____» _____ 2022 г.

Руководитель работы,
старший преподаватель
_____ Шелудько А.С.
«____» _____ 2022 г.

Челябинск 2022

1 ЗАДАНИЕ

1. Написать программу для создания множества точек и построения отрезка между двумя наиболее удаленными друг от друга точками множества. Для задания координат точек использовать генератор псевдослучайных чисел. Интерфейс программы должен содержать следующие элементы управления:

- создание множества точек;
- построение решеия;
- сохранение результата в файл;
- выход из программы.

2 МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

Пусть x_0, y_0, w, h – соответственно координаты левого верхнего угла, ширина и высота прямоугольной области. При генерации точек мы выбираем координаты в диапозоне x_0+50 $<$ x $<$ w-50 и y_0+50 $<$ y $<$ h-50. Так же координаты должны быть (x $>$ 250 и y $<$ 260) или (85 $<$ x $>$ 230 и y $<$ 130) чтобы не попасть в лес или озеро на карте.

Перебирая все точки находим самые дальние вычисляя расстояние между ними по формуле $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

3 ТЕКСТ ПРОГРАММЫ

Файл main.cpp

```
#include "graphics.h"
#include "control.h"
#include "task.h"

using namespace std;

int main()
{
   initwindow(600, 450);

   create_control(FILL_1,   0, 400, "create.bmp");
   create_control(FILL_2,  150, 400, "treat.bmp");
   create_control(SAVE,   300, 400, "save.bmp");
   create_control(EXIT,   450, 400, "exit.bmp");

   int left = 0, top = 0, width = 600, height = 400;
   Point points[15];
   IMAGE *image;

   image = loadBMP("Map.bmp");
   putimage(left, top, image, COPY_PUT);

   freeimage(image);
   while (true)
   {
      while(mousebuttons() != 1);
      switch(select_control())
      {
         case NONE: break;
         case FILL_1: creatPoint(points, left, top, width, height); b
         case FILL_2: treat(points); break;
         case SAVE: save(); break;
         case EXIT: closegraph(); return 0;
      }
      while(mousebuttons());
   }
}
```

Файл task.h

```
#ifndef TASK_H
#define TASK_H

#define COLOR_MAX 255

struct Point
{
   int x = 0, y = 0;
};

void creatPoint(Point[15], int, int, int, int);
void treat(Point[15]);
void save();

#endif
```

Файл task.cpp

```
#include <math.h>
#include <random>
#include <time.h>
#include "graphics.h"
#include "task.h"

void creatPoint(Point points[15], int left, int top, int width, int h
{
   IMAGE *image;

   image = loadBMP("Map.bmp");
   putimage(left, top, image, COPY_PUT);

   freeimage(image);
   srand(time(0));

   left += 50;
   top += 50;
   width -= 50;
   height -= 50;
   for(int i = 0; i < 15; i++)
   {
      points[i].x = rand()%(width-left)+left;
      points[i].y = rand()%(height-top)+top;
      if(points[i].x < 250 && points[i].y > 260 || points[i].x < 230
      {
         i--;
         continue;
      }
```

```
        setfillstyle(SOLID_FILL, BLACK);
        setcolor(BLACK);
        fillellipse(points[i].x, points[i].y, 4, 4);
    }
}

void treat(Point points[15])
{
    double l_max = 0, l;
    int twoPoint[2];
    for(int i = 0; i < 15; i++)
    {
        for(int k = i; k < 15; k++)
        {
            int x = points[i].x-points[k].x;
            int y = points[i].y-points[k].y;
            l = sqrt(x*x+y*y);
            if(l > l_max)
            {
                l_max = l;
                twoPoint[0] = i;
                twoPoint[1] = k;
            }
        }
    }
    line(points[twoPoint[0]].x, points[twoPoint[0]].y, points[twoPoint
}

void save()
{
    int width, height;
    IMAGE *output;

    width  = getmaxx() + 1;
    height = getmaxy() + 1;
    output = createimage(width, height);

    getimage(0, 0, width - 1, height - 1, output);
    saveBMP("output.bmp", output);
    freeimage(output);
}
```

Файл control.h

```
#ifndef CONTROL_H
#define CONTROL_H

enum control_values { NONE = -1, EXIT, SAVE,
                      FILL_1, FILL_2, FILL_3, FILL_4,
                      FILL_5, FILL_6, FILL_7, FILL_8,
                      N_CONTROLS };
```

```
struct Control
{
   int left;
   int top;
   int right;
   int bottom;
};

void create_control(int, int, int, const char*);
int select_control();

#endif
```

Файл control.cpp

```
#include "graphics.h"
#include "control.h"

Control controls[N_CONTROLS];

void create_control(int i, int left, int top,
                    const char *file_name)
{
   IMAGE *image;

   image = loadBMP(file_name);
   putimage(left, top, image, COPY_PUT);

   controls[i].left   = left;
   controls[i].top    = top;
   controls[i].right  = left + imagewidth(image) - 1;
   controls[i].bottom = top  + imageheight(image) - 1;

   freeimage(image);
}

int select_control()
{
   int x, y;

   x = mousex();
   y = mousey();

   for (int i = 0; i < N_CONTROLS; i++)
   {
      if (x > controls[i].left && x < controls[i].right &&
          y > controls[i].top  && y < controls[i].bottom)
      {
         return i;
      }
```

```
    }

    return NONE;
}
```

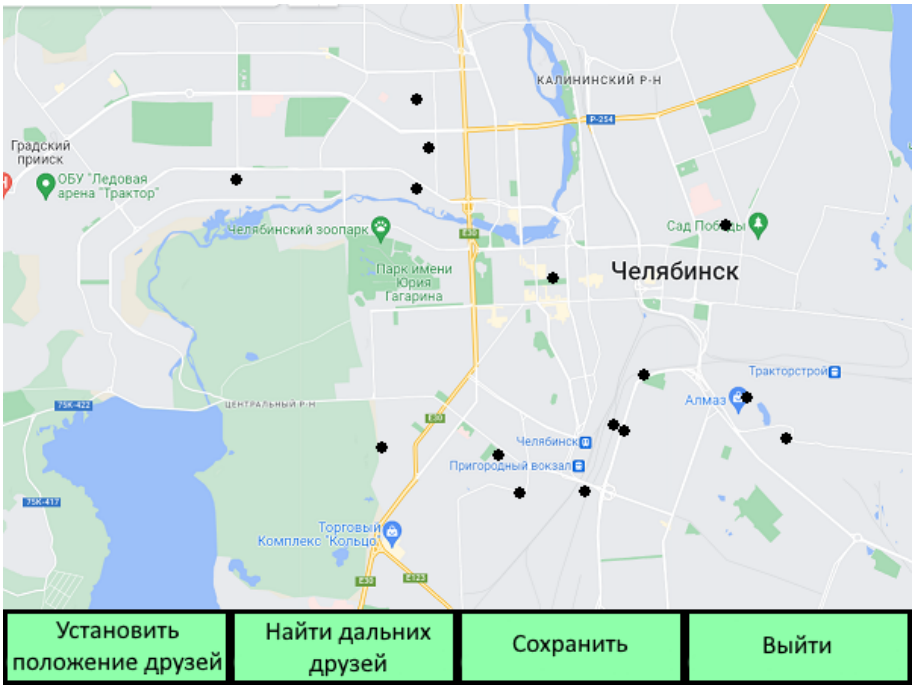

Рисунок 4.1 – Результат выполнения программы (функция creatPoint, кнопка "Установить положение друзей")

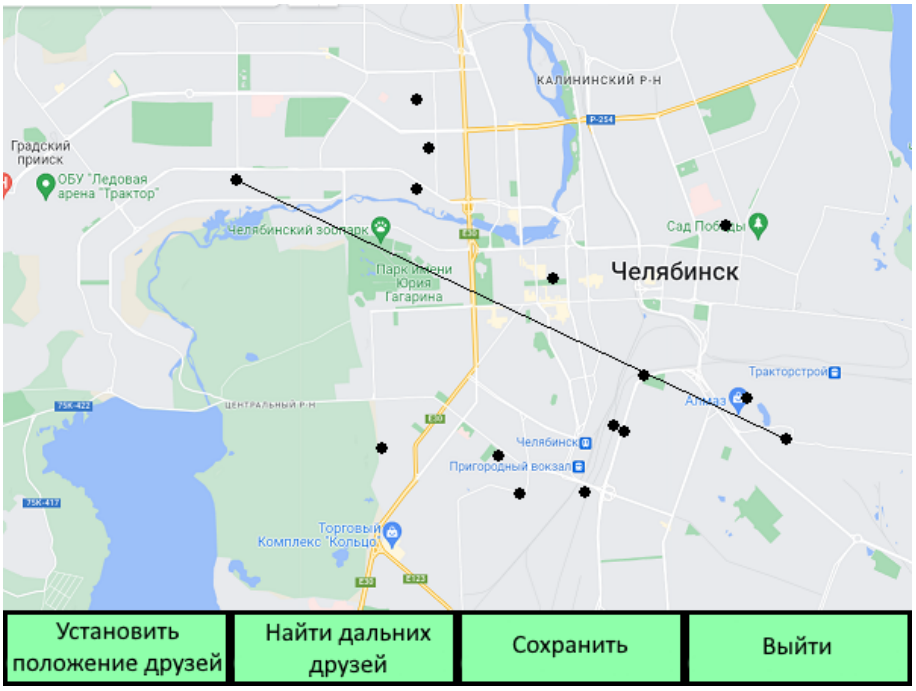

Рисунок 4.2 – Результат выполнения программы (функция treat, кнопка "Найти дальних друзей")