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 № 1 по дисциплине
«Математические основы компьютерной графики»

Автор работы,
студент группы ЕТ-212
_____ Олейников Е.О.
«____» _____ 2022 г.

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

Челябинск 2022

1 ЗАДАНИЕ

1. Разработать подпрограммы для градиентной закраски прямоугольной области различными способами. Аргументы подпрограмм – координаты левого верхнего угла, ширина и высота области. Подпрограммы должны работать для любого соотношения размеров прямоугольной области. Для каждого способа закраски определить зависимость интенсивности цвета от координат точки. При реализации подпрограмм использовать для рисования только процедуру putpixel.

2. Написать программу для тестирования разработанных подпрограмм. Интерфейс программы должен содержать следующие элементы управления:

- выбор способа закраски;
- 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 закрашивании выполняется обход всех точек области. Цвет точки $(x; y)$ задается в цветовой модели RGB. Значения интенсивностей равны 0, $\frac{c}{2}$ и c соответственно для красной, зеленой и синей составляющей цвета. Максимальное значение интенсивности $c_{max} = 255$. Переменная c, определяющая интенсивность составляющих цвета, вычисляется в соответствии с приведенными ниже формулами.

Способ 1. Интенсивность цвета зависит только от ординаты точки:

$$c = \frac{y - y_0}{h} c_{max}.$$

Способ 7. Интенсивность цвета зависит от расстояния от точки до центра прямоугольной области:

$$c = \begin{cases} (1 - d)c_{max}, d < 1; \\ 0, d \geq 1, \end{cases}$$

где

$$d = \frac{1}{r}\sqrt{\left(x - x_0 - \frac{w}{2}\right)^2 + \left(y - y_0 - \frac{h}{2}\right)^2},$$

$$r = \begin{cases} \frac{w}{2}, w < h; \\ \frac{h}{2}, w \geq h. \end{cases}$$

3 ТЕКСТ ПРОГРАММЫ

Файл main.cpp

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

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

   create_control(FILL_1,   0, 400, "fill_1.bmp");
   create_control(FILL_2,  50, 400, "fill_2.bmp");
   create_control(FILL_3, 100, 400, "fill_3.bmp");
   create_control(FILL_4, 150, 400, "fill_4.bmp");
   create_control(FILL_5, 200, 400, "fill_5.bmp");
   create_control(FILL_6, 250, 400, "fill_6.bmp");
   create_control(FILL_7, 300, 400, "fill_7.bmp");
   create_control(FILL_8, 350, 400, "fill_8.bmp");
   create_control(SAVE,   400, 400, "save.bmp");
   create_control(EXIT,   500, 400, "exit.bmp");

   int left = 0, top = 0, width = 600, height = 400;

   while (true)
   {
      while (mousebuttons() != 1);
      switch (select_control())
      {
         case NONE: break;
         case FILL_1: fill_1(left, top, width, height); break;
         case FILL_2: fill_2(left, top, width, height); break;
         case FILL_3: fill_3(left, top, width, height); break;
         case FILL_4: fill_4(left, top, width, height); break;
         case FILL_5: fill_5(left, top, width, height); break;
         case FILL_6: fill_6(left, top, width, height); break;
         case FILL_7: fill_7(left, top, width, height); break;
         case FILL_8: fill_8(left, top, width, height); break;
         case SAVE: save(); break;
         case EXIT: closegraph(); return 0;
      }
   }
}
```

Файл task.h

```
#ifndef TASK_H
#define TASK_H

#define COLOR_MAX 255

void fill_1(int, int, int, int);
void fill_2(int, int, int, int);
void fill_3(int, int, int, int);
void fill_4(int, int, int, int);
void fill_5(int, int, int, int);
void fill_6(int, int, int, int);
void fill_7(int, int, int, int);
void fill_8(int, int, int, int);
void save();

#endif
```

Файл task.cpp

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

void fill_1(int left, int top, int width, int height)
{
   double x, y;
   int c;

   for (x = 0; x < width; x++)
   {
      for (y = 0; y < height; y++)
      {
         c = y / height * COLOR_MAX;
         putpixel(left + x, top + y, COLOR(0, c / 2, c));
      }
   }
}

void fill_2(int left, int top, int width, int height)
{

}

void fill_3(int left, int top, int width, int height)
{

}
```

```
void fill_4(int left, int top, int width, int height)
{

}

void fill_5(int left, int top, int width, int height)
{

}

void fill_6(int left, int top, int width, int height)
{

}

void fill_7(int left, int top, int width, int height)
{
   double x, y, d, r;
   int c;

   r = (width > height ? height : width) / 2;

   for (x = 0; x < width; x++)
   {
      for (y = 0; y < height; y++)
      {
         d = sqrt((x - width / 2) * (x - width / 2) +
                  (y - height / 2) * (y - height / 2)) / r;
         c = d < 1 ? (1 - d) * COLOR_MAX : 0;
         putpixel(left + x, top + y, COLOR(0, c / 2, c));
      }
   }
}

void fill_8(int left, int top, int width, int height)
{

}

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 (способ 1)

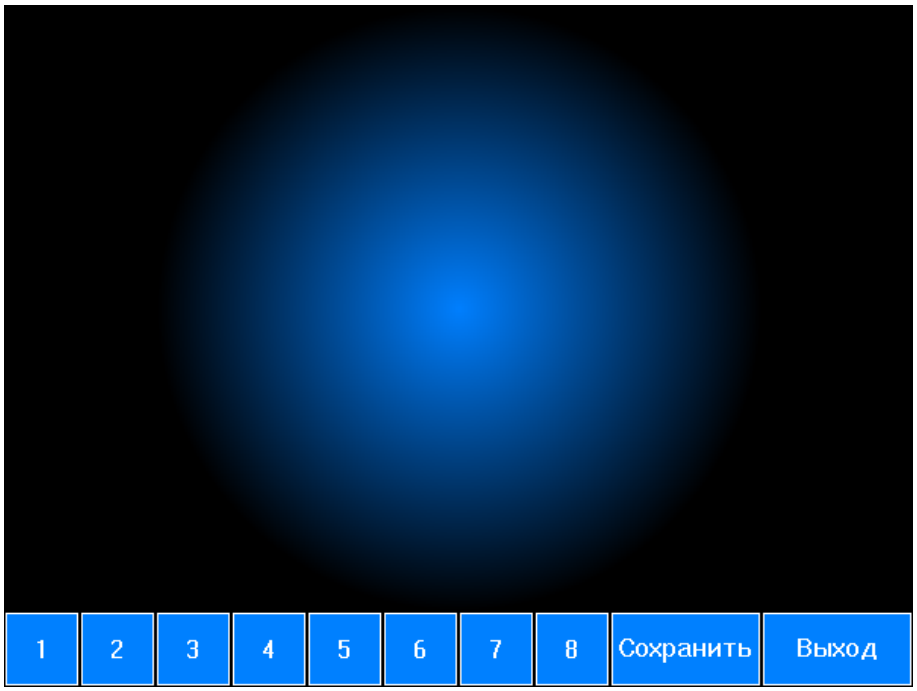

Рисунок 4.2 – Результат выполнения программы (способ 7)