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 № 4 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. Привести описание и схему алгоритма Брезенхема для растрового представления окружности.

2. Разработать подпрограмму для рисования окружности (аналог процедуры circle из графической библиотеки). Аргументы подпрограммы – координаты центра и радиус окружности. При реализации подпрограммы использовать для рисования только процедуру putpixel. Для определения текущего цвета рисования использовать функцию getcolor.

3. Разработать подпрограмму для создания фигуры, показанной на рисунке. При создании контура фигуры использовать собственную подпрограмму рисования окружности. Для закраски деталей фигуры использовать процедуру floodfill.

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

- изменение цвета деталей и фона;
- сохранение результата в файл;
- выход из программы.

2 МАТЕМАТИЧЕСКАЯ МОДЕЛЬ

Пусть x_1, y_1, x_2, y_2 – соответственно координаты центра окружности. При рисовании линии цвет задан изначально color через getcolor(), а x и y изменения координат пикселей.

Рисование окружности:

$$x = 0 \quad y = 0.$$

$$delta = 1 - 2 * r.$$

$$err = 0.$$

Пока y $\geq$ x выполняем следующие действия: Ставим пиксели в точки: (x_1+x, y_1+y), (x_1+x, y_1-y), (x_1-x, y_1+y), (x_1-x, y_1-y), (x_1+y, y_1+x), (x_1+y, y_1-x), (x_1-y, y_1+x), (x_1-y, y_1-x).

$$err = 2 * (delta + y) - 1.$$

Если delta $<$ 0 и err $\leq$ 0, то:

$$delta = delta + 2 * x + 1.$$

$$x = x + 1.$$

Если delta $>$ 0 и err $>$ 0, то:

$$delta = delta - 2 * y + 1.$$

$$y = y - 1.$$

В остальных случаях:

$$delta = delta + 2 * (x - y).$$

$$y = y - 1 \quad x = x + 1.$$

3 ТЕКСТ ПРОГРАММЫ

Файл main.cpp

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

#include "control.h"
#include "task.h"

using namespace std;

struct Circl
{
   int x, y, r, thickness;
};

struct Pole
{
   int x, y, value;
};

struct Button
{
   int x, y;
   IMAGE *image;
};

int main()
{
   setlocale(LC_ALL,"Russian");
   setlocale(LC_ALL,"rus");
   srand(time(0));
   initwindow(850, 700);

   setlinestyle(SOLID_LINE, 0, 1);
   setbkcolor(WHITE);

   srand(time(0));

   IMAGE *imager;
   imager = loadBMP("Red.bmp");

   IMAGE *imageg;
   imageg = loadBMP("Green.bmp");

   IMAGE *imageb;
   imageb = loadBMP("Blue.bmp");
```

```
int col[3][2] = {{200, 0}, {200, 0}, {200, 0}};
int numCircle = 0, wrt[4] = {0};
int fil = 0, p = 0, t = 0;
int fillPoint[1000][3] = {0};
int numFill = 0;

Circl data[100];
Pole r = {40, 660, 100};
Pole thickness = {190, 660, 1};

IMAGE *image;
image = loadBMP("fill.bmp");
Button BFill = {640, 660, image};
image = loadBMP("circle.bmp");
Button BCircle = {640, 660, image};
image = loadBMP("clear.bmp");
Button BClear = {690, 660, image};
image = loadBMP("save.bmp");
Button BSave = {740, 660, image};
image = loadBMP("exit.bmp");
Button BExit = {790, 660, image};

while(1)
{
   if(1-fil)
   {
      p = 1-p;
      setactivepage(p);

      setfillstyle(SOLID_FILL, BLACK);
      setcolor(WHITE);
      rectangle(-1, -1, 851, 701);
      bar(0, 0, 850, 700);

      if(mousebuttons() and mousey()+r.value < 650)
      {
         data[numCircle] = {mousex(), mousey(), r.value, thickness
         numCircle++;
         while(mousebuttons());
      }

      if(mousebuttons() && abs(r.x+75-mousex()) < 75 && abs(r.y+13
      {
         for(int i = 0; i < 2; i++) wrt[i] = 0;
         wrt[0] = 1;
      }
      if(mousebuttons() && abs(thickness.x+75-mousex()) < 75 && ab
      {
         for(int i = 0; i < 2; i++) wrt[i] = 0;
         wrt[1] = 1;
      }
```

```
setfillstyle(SOLID_FILL, COLOR(100, 100, 100));
bar(0, 650, 850, 700);

circleBre(360, 675, 20, 1);
setfillstyle(SOLID_FILL, COLOR(255, 0, 0));
floodfill(360, 675, WHITE);
circleBre(400, 675, 20, 1);
setfillstyle(SOLID_FILL, COLOR(255, 255, 0));
floodfill(400, 675, WHITE);
circleBre(440, 675, 20, 1);
setfillstyle(SOLID_FILL, COLOR(0, 255, 0));
floodfill(440, 675, WHITE);
circleBre(480, 675, 20, 1);
setfillstyle(SOLID_FILL, COLOR(0, 255, 255));
floodfill(480, 675, WHITE);
circleBre(520, 675, 20, 1);
setfillstyle(SOLID_FILL, COLOR(0, 0, 255));
floodfill(520, 675, WHITE);
circleBre(560, 675, 20, 1);
setfillstyle(SOLID_FILL, COLOR(255, 0, 255));
floodfill(560, 675, WHITE);
circleBre(600, 675, 20, 1);
setfillstyle(SOLID_FILL, COLOR(255, 255, 255));
floodfill(600, 675, WHITE);

r.value = pole(r.x, r.y, 130, 25, "???????: ", r.value, wrt[
thickness.value = pole(thickness.x, thickness.y, 130, 25, "?

setcolor(WHITE);
for(int i = 0; i < numCircle; i++) circleBre(data[i].x, data

if(button(BClear))
{
   numFill = 0;
   numCircle = 0;
   while(mousebuttons());
}
if(button(BSave))
{
   save();
   while(mousebuttons());
}
if(button(BFill))
{
   fil = 1;
   while(mousebuttons());
}
if(button(BExit)) return 0;

t = t+1;
t = t%8;
```

```
        setvisualpage(p);
        delay(10);
    }
    else
    {
        p = 1-p;
        setactivepage(p);

        setfillstyle(SOLID_FILL, BLACK);
        setcolor(WHITE);
        rectangle(-1, -1, 851, 701);
        bar(0, 0, 850, 700);

        if(mousebuttons() and mousey() < 650)
        {
            fillPoint[numFill][0] = mousex();
            fillPoint[numFill][1] = mousey();
            fillPoint[numFill][2] = COLOR(col[0][0], col[1][0], col[2
            numFill++;
        }

        setcolor(WHITE);
        for(int i = 0; i < numCircle; i++) circleBre(data[i].x, data
        for(int i = 0; i < numFill; i++)
        {
            setfillstyle(SOLID_FILL, fillPoint[i][2]);
            floodfill(fillPoint[i][0], fillPoint[i][1], WHITE);
        }

        setfillstyle(SOLID_FILL, COLOR(100, 100, 100));
        bar(0, 650, 850, 700);
        lineColor(10, col, 0, imager);
        lineColor(220, col, 1, imageg);
        lineColor(430, col, 2, imageb);

        if(button(BCircle))
        {
            fil = 0;
            while(mousebuttons());
        }
        if(button(BSave))
        {
            save();
            while(mousebuttons());
        }
        if(button(BClear))
        {
            numFill = 0;
            while(mousebuttons());
        }
        if(button(BExit)) return 0;
```

```
            t = t+1;
            t = t%8;

            setvisualpage(p);
            delay(10);
        }
    }

    getch();
}
```

Файл task.h

```
#ifndef TASK_H
#define TASK_H

void circleBre(int, int, int, int);
void fill(int, int, int[3][2]);
void save();

#endif
```

Файл task.cpp

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

using namespace std;

void circleBre(int x1, int y1, int r, int thickness)
{
   int color = getcolor();
   for(int i = 1; i < thickness+1; i++)
   {
      int x = 0;
      int y = r+i;
      int delta = 1-2*(r+i/2*pow(-1, i-1));
      int err = 0;
      while(y >= x)
      {
         putpixel(x1+x, y1+y, color);
         putpixel(x1+x, y1-y, color);
         putpixel(x1-x, y1+y, color);
         putpixel(x1-x, y1-y, color);
         putpixel(x1+y, y1+x, color);
         putpixel(x1+y, y1-x, color);
         putpixel(x1-y, y1+x, color);
         putpixel(x1-y, y1-x, color);
         err = 2*(delta+y)-1;
         if(delta < 0 && err <= 0)
         {
            delta += 2*++x+1;
            continue;
         }
         if(delta > 0 && err > 0)
         {
            delta -= 2*--y+1;
            continue;
         }
```

```
            delta += 2*(++x- --y);
        }
    }
}

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

struct Button;

int pole(int, int, int, int, std::string, int, int&, int);
void lineColor(int, int[3][2], int, IMAGE*);
bool button(Button);

#endif
```

Файл control.cpp

```
#include <string>
#include <graphics.h>

#include "control.h"

using namespace std;

struct Button
{
    int x, y;
    IMAGE *image;
};

int pole(int left, int top, int widht, int height, string Text, int d
{
    if(wrt)
```

```
    {
        if(kbhit())
        {
            int key = getch();
            if(key == 13) wrt = 0;
            if(key == 8) data = data/10;
            if(key > 47 && key < 58) data = data*10+key-48;
        }

        setfillstyle(SOLID_FILL, WHITE);
        bar(left, top, left+widht, top+height);
        setcolor(BLACK);
        if(t > 3)
        {
            char text[30];
            sprintf(text, "%s%d", Text.c_str(), data);
            outtextxy(left+2, top+2, text);
        }
        else
        {
            char text[30];
            sprintf(text, "%s%d|", Text.c_str(), data);
            outtextxy(left+2, top+2, text);
        }
    }
    else
    {
        setfillstyle(SOLID_FILL, WHITE);
        bar(left, top, left+widht, top+height);
        setcolor(BLACK);
        char text[30];
        sprintf(text, "%s%d", Text.c_str(), data);
        outtextxy(left+2, top+2, text);
    }
    return data;
}

void lineColor(int x, int col[3][2], int num, IMAGE *image)
{
    putimage(x, 660, image, COPY_PUT);

    setfillstyle(SOLID_FILL, WHITE);
    bar(col[num][0]*200/255+x-3, 660, col[num][0]*200/255+x+3, 689);

    if(mousebuttons())
    {
        if(abs(mousex()-col[num][0]*200/255-x) < 4 and abs(mousey() - 6
        {
            col[num][1] = 1;
        }
    }
    else
```

```
    {
        col[num][1] = 0;
    }

    if(col[num][1])
    {
        col[num][0] = (mousex()-x)*255/200;
        col[num][0] = (col[num][0]+abs(col[num][0]))/2;
        col[num][0] = col[num][0]>255 ? 255 : col[num][0];
    }
}

bool button(Button a)
{
    putimage(a.x, a.y, a.image, COPY_PUT);
    return abs(a.x+25-mousex()) < 25 && abs(a.y+15-mousey()) < 15 && m
}
```

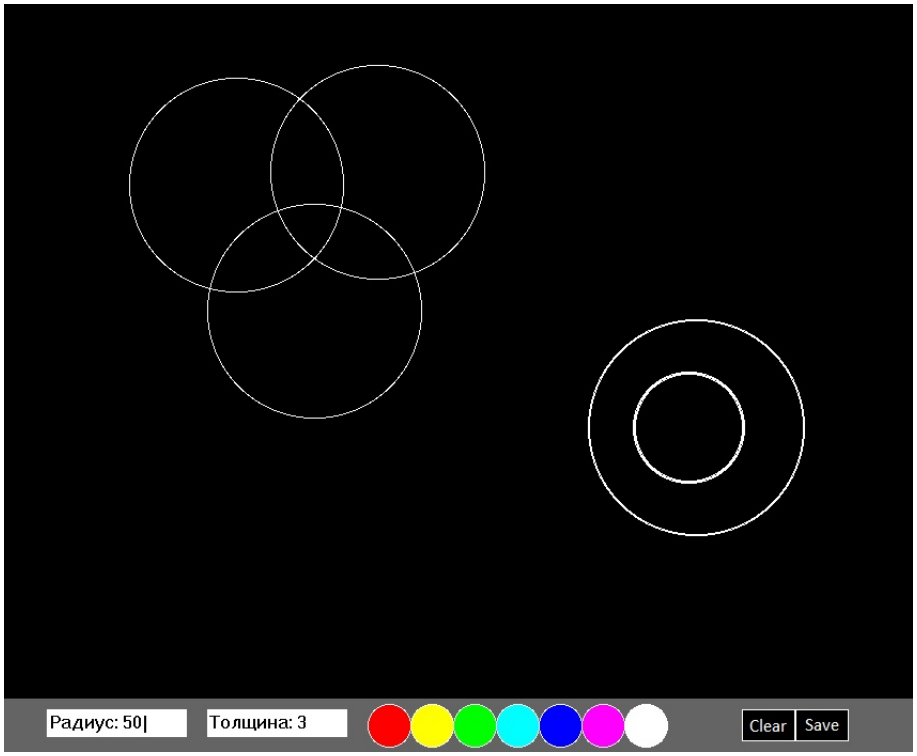

Рисунок 4.1 – Результат выполнения программы (Пример 1)

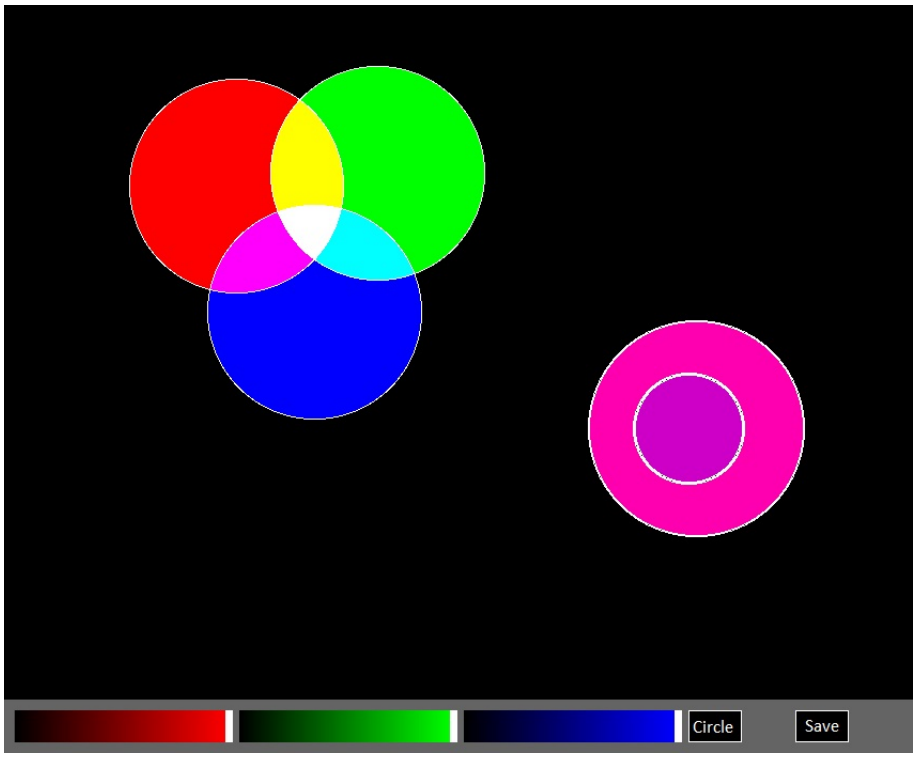

Рисунок 4.2 – Результат выполнения программы (Пример 2)