ФГАОУВО «Южно-Уральский государственный университет (НИУ)» Институт естественных и точных наук

Кафедра «Прикладная математика и программирование»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

по дисциплине «Объектно-ориентированное программирование»

Автор работы

студент группы ЕТ-213

Ильюп В.А.

2022 г.

Работа зачтена с оценкой

А.К. Демидов

2022 г.

Челябинск, 2022

# 1 Постановка задачи

I. Базовый класс для всех вариантов:

class Figure

{

int c; // цвет

bool visible;

protected:

int x,y; // базовая точка

virtual void draw();

public:

Figure(int c, int x, int y);

~Figure();

void move(int x, int y); // сместить фигуру в точку (x,y)

// видимая фигура гасится, затем рисуется в другом месте

// у невидимой просто меняются поля x,y

void setcolor(int c); // установить цвет фигуры

// видимая фигура рисуется новым цветом

// у невидимой просто меняется поле c

int getcolor() const; // получить цвет

void hide(); // спрятать: нарисовать черный прямоугольник

// по размерам area()

void show(); // показать

bool isvisible() const; // видима?

virtual void area(int &x1,int &y1,int &x2,int &y2) const;

// получить размеры прямоугольной области, содержащей фигуру

};

Определить реализацию методов класса Figure.  
Методы area и draw нужно определить как чисто виртуальные.  
Как нужно определить деструктор Figure и производных классов, чтобы видимый объект исчезал с экрана при уничтожении?  
Определить производный класс

12. Пятиугольная звезда   
Pentagram(цвет линий, x и y центра, радиус, угол поворота)

Определить дополнительный метод в производном классе для изменения размеров:  
    void setsizes(длина, высота);  
или void setsizes(длина, высота, радиус);  
или void setsizes(радиус, угол1, угол2);  
и т.д., т.е. изменение значений, указываемых в аргументах конструтора, начиная с четвертого.  
  
От написанного класса произвести новый дочерний класс - закрашенная фигура.  
Например, закрашенный ромб (FillRomb ← Romb ← Figure).  
Добавить к параметрам конструктора цвет заполнения.  
Определить дополнительный метод для изменения цвета заполнения:  
void setfillcolor(int c);  
  
II. Реализовать main c тестами  
Динамически создать две фигуры 2 разных классов, адреса объектов сохранить в переменных типа Figure \*. Вызвать все методы для каждой из фигур, перед вызовом методов, определенных в производных классах, выполнить преобразование к указателю на производный класс с помощью dynamic\_cast с проверкой:  
if(Romb \*r=dynamic\_cast<Romb\*>(o1)) r->setsizes(100,50);

# 2 Описание интерфейса класса

class Figure

{

int c; // цвет

bool visible; // состояние видимости protected:

int x, y; // базовая точка virtual void draw() const = 0;

public:

Figure(int c, int x, int y) : c(c), x(x), y(y), visible(false) { } // конструктор virtual ~Figure() { } // виртуальный деструктор для возможности удаления

объекта через указатель на базовый класс

void move(int x, int y); // переместить фигуру в координату (x, y) void setcolor(int c); // задать новый цвет

inline int getcolor() const { return c; } // получить цвет void hide(); // спрятать

void show(); // показать

bool isvisible() const { return visible; } // видима фигура в данный момент virtual void area(int& x1, int& y1, int& x2, int& y2) const = 0;

};

class Pentagram : public Figure{ protected:

int R,alpha;//радиус и угол поворота void draw() const;//метод рисования

public:

Pentagram(int c, int x, int y, int R, int alpha): R(R), alpha(alpha), Figure(c, x, y){}//конструктор

~Pentagram() {hide();}//деструктор

//далее идут геттеры int getx(){return x;} int gety(){return y;} int getR(){return R;}

int getalpha(){return alpha;}

//метод изменения размеров

void setsizes(int x, int y, int R, int alpha);

//получить размеры прямоугольной области, содержащей фигуру void area(int& x0, int& y0, int& x1, int& y1) const;

};

class FillPentagram : public Pentagram{ private:

int backcolor;//цвет фона protected:

void draw() const;//метод рисования с закрашенным фоном public:

FillPentagram(int c, int backcolor, int x, int y, int R, int alpha):backcolor(backcolor), Pentagram(c, x, y, R, alpha){}//конструктор

~FillPentagram(){}//деструктор

void setfillcolor(int c);//задать цвет фона

int getfillcolor() const{return backcolor;}//получить цвет фона

};

# 3 Описание тестов для проверки классов

int main(){ initwindow(800,600,"Laba3");

//создаем пентаграмму

Figure\* f = new Pentagram(RED, 400, 300, 100, 60); Pentagram\* ff = dynamic\_cast<Pentagram\*>(f);

//проверяем работу геттеров

cout << "Цвет(число от 1 до 16): " << f->getcolor() << endl;

cout << "Координаты х и у соответственно: " << ff->getx() << ", " << ff-

>gety() << endl;

cout << "Радиус и угол поворота: " << ff->getR() << ", " << ff->getalpha() << endl;

// выводим координаты противоположных точек прямоугольной области int x1, y1, x2, y2;

ff->area(x1, y1, x2, y2);

cout << x1 << ' ' << y1 << endl; cout << x2 << ' ' << y2 << endl; getch();

f->show();

getch();

//скрываем её

if(f->isvisible())f->hide(); getch();

//проверка основных методов f->setcolor(WHITE);

f->move(200,200);

f->show();

getch();

f->setcolor(GREEN);

ff->setsizes(100,100,50,60); ff->show();

if(ff->isvisible())ff->hide(); getch();

Figure\* z = new FillPentagram(RED,YELLOW,400,300,200, 30); FillPentagram\* fff = dynamic\_cast<FillPentagram\*>(z);

fff->show();

getch();

if(fff->isvisible())fff->hide(); fff->setcolor(BROWN);

fff->setfillcolor(BLUE); getch();

fff->show();

getch();

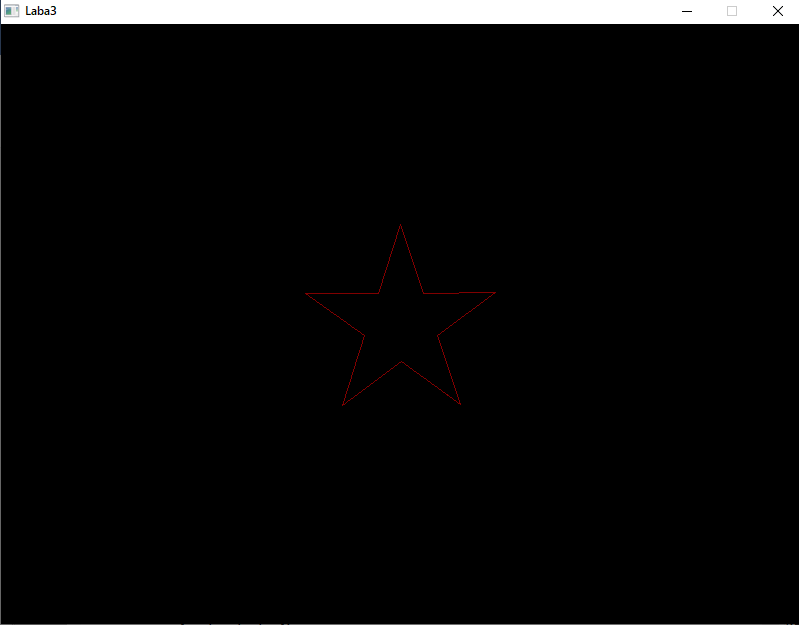
fff->hide();

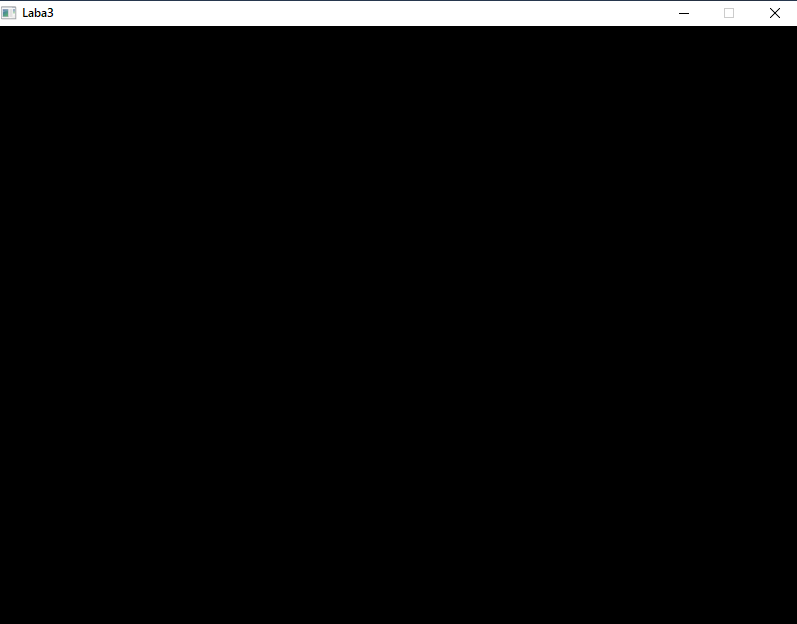
fff->setsizes(300,200,100,50); fff->show();

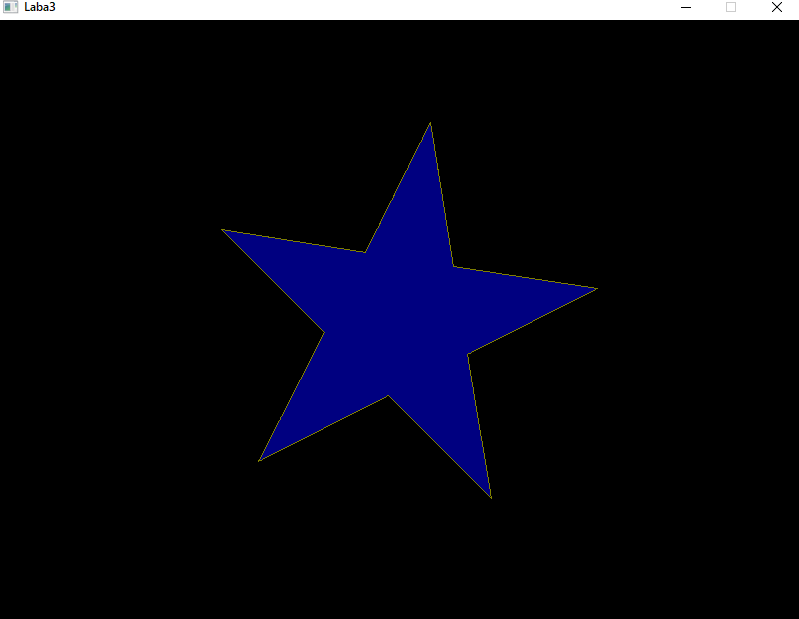
getch(); return 0;

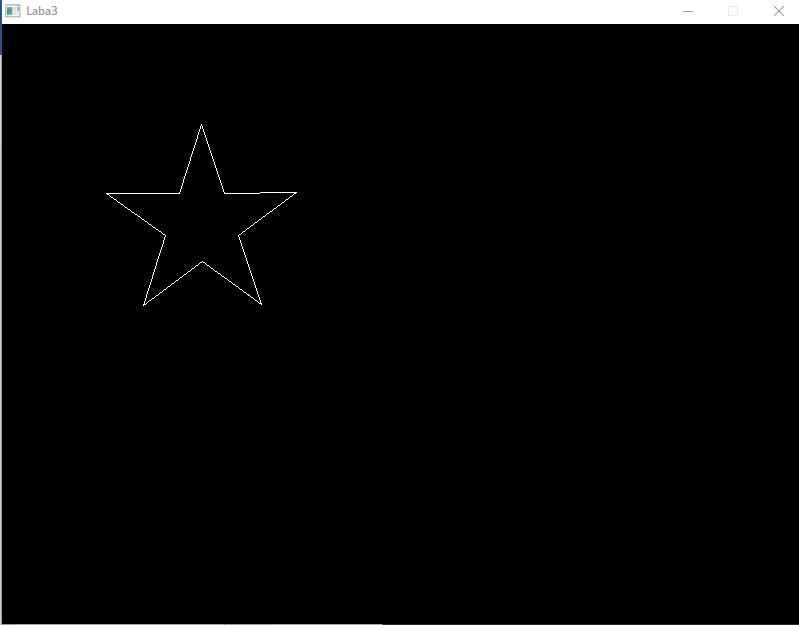
}

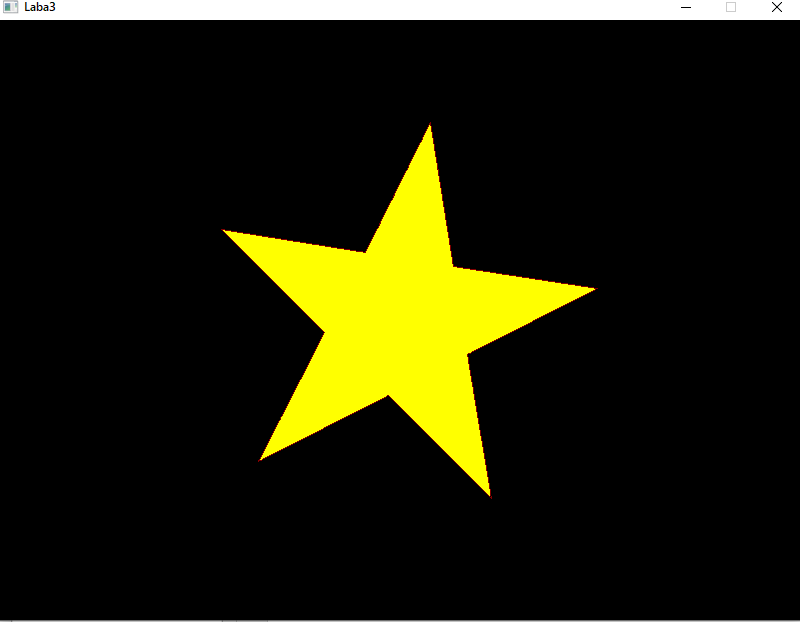
**Полученные результаты:**

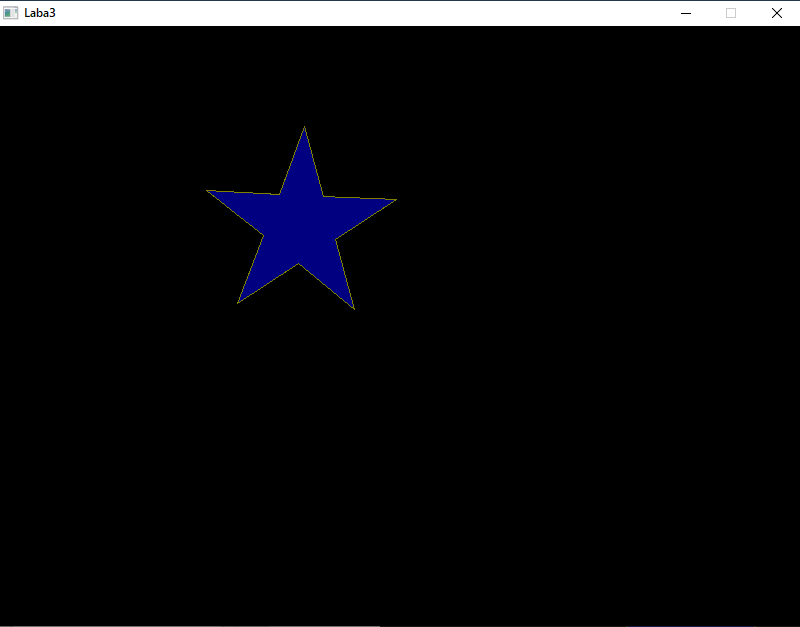












# 4 Листинг реализации класса

void Pentagram::area(int& x0, int& y0, int& x1, int& y1) const{ x0 = x - R;

y0 = y - R; x1 = x + R; y1 = y + R;

}

void Pentagram::setsizes(int xf, int yf, int Rf, int alphaf){ bool v = isvisible();

hide(); x = xf; y = yf; R = Rf;

alpha = alphaf; if(v) show();

}

void Pentagram::draw() const{

::setcolor(getcolor());

double z = acos(-1) / 5;

double r = (R\*cos(2 \* z))/cos(z); for (int i = 0; i <= 10 ; i++){

int l = (i % 2) ? r : R;

double x1 = (x + l\*cos(i\*z + alpha)); double y1 = (y + l\*sin(i\*z + alpha)); if (!i) moveto (x1, y1);

lineto(x1, y1); moveto (x1, y1);

}

}

void FillPentagram::setfillcolor(int backcolor{ this->backcolor = backcolor;

if(isvisible()) draw();

}

void FillPentagram::draw()const{

::setcolor(this->getcolor());

::setfillstyle(SOLID\_FILL, backcolor); Pentagram::draw();

floodfill(x,y,::getcolor());

}

void FillPentagram::setfillcolor(int backcolor){ this->backcolor = backcolor;

if(isvisible()) draw();

}

void FillPentagram::draw()const{

::setcolor(this->getcolor());

::setfillstyle(SOLID\_FILL, backcolor); Pentagram::draw();

floodfill(x,y,::getcolor());

}