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-212

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Александрова

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

Работа зачтена с оценкой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.К. Демидов

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

Челябинск, 2017

# 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 dx, int dy); // сместить фигуру на

//(dx,dy)

// 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(); // получить цвет

void hide(); // спрятать: нарисовать черный

//прямоугольник

// по размерам area()

void show(); // показать

bool isvisible(); // видима?

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

// 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

1. Ромб  
Romb(цвет линий, x и y центра, длина, высота)

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;

# protected:

# int x, y; //базовая точка

# virtual void draw() = 0;

# public:

# Figure(int c, int x, int y):c(c), visible(0), x(x), y(y) {}

# virtual ~Figure() {}

# void move(int dx, int dy); //сместить фигуру на dx, dy

# void setcolor(int c); //установить цвет фигуры, видимая рисуется, у //невидимой меняется цвет

# int getcolor() const { return c; } //получить цвет

# void hide(); //спрятать

# void show(); //показать

# bool isvisible() const { return visible; } //видима?

# virtual void area(int &x1, int &y1, int &x2, int &y2) const = 0; //размеры //обрасти, содержащей фигуру

# };

# class Romb: public Figure {

# protected:

# int l, h; //длина и высота ромба

# void draw();

# public:

# Romb(int c, int x, int y, int l, int h): Figure(c, x, y), l(l), h(h) {}

# ~Romb() { hide(); }

# void setsizes(int l, int h); //изменение длины и высоты ромба

# void area(int &x1, int &y1, int &x2, int &y2)const; //область, где нарисована фигура

# };

# class fRomb: public Romb {

# protected:

# int fc; //Цвет

# void draw();

# public:

# fRomb(int c, int x, int y, int l, int h, int fc):Romb(c, x, y, l, h), fc(fc) {}

# void setfcolor(int);//изменить цвет

# };

# 3 Описание тестов для проверки классов

int main() {

initwindow(640, 480);

Figure \*R=new Romb(GREEN, 110, 110, 157, 112);

Figure \*R1=new fRomb(YELLOW, 200, 300, 100, 75, BROWN);

R->show();

R1->show();

getch();

R->hide();

R1->hide();

getch();

R->move(90, 90);

R1->move(75, 115);

R->show();

R1->show();

getch();

R->setcolor(WHITE);

R1->setcolor(RED);

getch();

if(Romb \*r=dynamic\_cast<Romb\*>(R)) r->setsizes(100,30);

if(Romb \*r=dynamic\_cast<Romb\*>(R1)) r->setsizes(100,100);

getch();

if(fRomb \*r=dynamic\_cast<fRomb\*>(R)) r->setfcolor(GREEN);

if(fRomb \*r=dynamic\_cast<fRomb\*>(R1)) r->setfcolor(GREEN);

getch();

delete R;

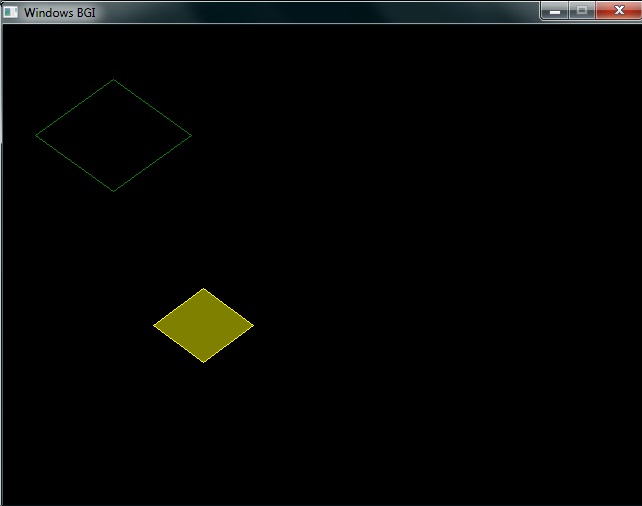
delete R1;

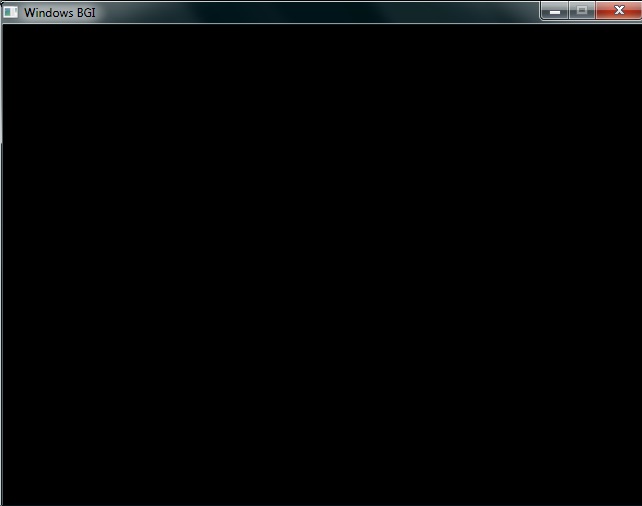
getch();

return 0;

}

**Полученные результаты:**

****

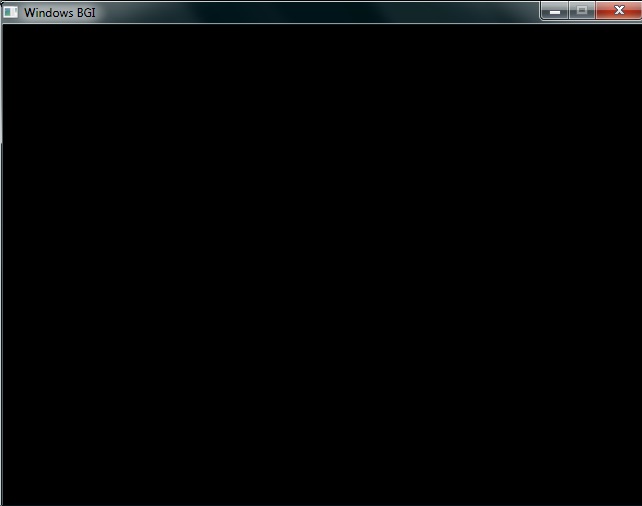
****











# 4 Листинг реализации класса

void Figure::setcolor(int c) {

this->c=c;

if (visible) draw();

}

void Figure::move(int dx, int dy) {

bool f=visible;

if (f) hide();

x+=dx;

y+=dy;

if (f) show();

}

void Figure::hide() {

if (visible==0) return;

int x1, y1, x2, y2;

area(x1, y1, x2, y2);

setfillstyle(SOLID\_FILL, BLACK);

bar(x1, y1, x2, y2);

visible=0;

}

void Figure::show() {

if (visible==1) return;

visible=1;

draw();

}

void Romb::setsizes(int l, int h) {

bool f=isvisible();

if (f) hide();

this->l=l;

this->h=h;

if (f) show();

}

void Romb::area(int &x1, int &y1, int &x2, int &y2)const {

x1=x-l/2;

y1=y-h/2;

x2=x+l/2;

y2=y+h/2;

}

void Romb::draw() {

::setcolor(getcolor());

moveto(x-l/2, y);

lineto(x,y-h/2);

…

lineto(x-l/2, y);

}

void fRomb::draw() {

int b[8]={x-l/2, y,

…

x,y+h/2

};

setfillstyle(SOLID\_FILL, fc);

fillpoly(4, b);

Romb::draw();

}

void fRomb::setfcolor(int a) {

fc=a;

if (isvisible()) draw();

}