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

_____ Б.А. Мухутдинов

_____2022 г.

Работа зачтена с оценкой

_____

_____А.В. Лут

_____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

14.Бублик
Boublik(цвет линий, x и y центра, радиус1, радиус2) радиус1<радиус2

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 с 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);
III. Написать отчет

- Постановка задачи
- Описание интерфейса классов (class {} и комментарии ко всем полям, методам и функциям)
- Описание тестов для проверки классов (main с комментариями, какие действия выполнялись, полученные результаты))
- Листинг реализации классов (реализация методов и функций)

2 Описание интерфейса класса

```
class Figure{

   int c; //цвет линий

   bool visible; //видимость

protected:

   int x, y; // базовая точка

   virtual void draw()=0; //нарисовать

public:

   Figure(int col, int x_, int y_):c(col), visible(0), x(x_),
y(y_) {}

   virtual ~Figure() {}

   void move(int x2, int y2); //сместить фигуру в (x2,y2)

   void setcolor(int col); // 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 области, содержащей фигуру

};


class Boublik: public Figure{
```

```
protected:

   int r1, r2;

   void draw();

public:

   Boublik(int col, int x_, int y_, int r_1, int r_2):
Figure(col, x_, y_), r1(r_1), r2(r_2) {}

   virtual ~Boublik() {hide();}

   void setsizes(int r_1, int r_2); //изменение внутренного и
внешнего радиуса бублика

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

};


class FillBoublik: public Boublik{

protected:

   int fc; //цвет закраски

   void draw(); //нарисовать

public:

   FillBoublik(int col, int x_, int y_, int r_1, int r_2, int
fillcol): Boublik(col, x_, y_, r_1, r_2), fc(fillcol){}

   void setfillcolor(int fillcol); //изменить цвет закраски

};
```

3 Описание тестов для проверки классов

```
int main(){

   initwindow(1000, 800);

   Figure *figure1 = new Boublik(WHITE, 200, 200, 30, 100);

   Figure *figure2 = new FillBoublik(BLUE, 500, 500, 50, 150,
LIGHTGREEN);

   figure1->show();

   figure2->show();

   getch();

   figure1->hide();
```

```
   figure2->hide();

   getch();

   figure1->move(600, 500);

   figure2->move(200, 250);

   figure1->show();

   figure2->show();

   getch();

   figure1->setcolor(LIGHTBLUE);

   figure2->setcolor(WHITE);

   // проверяем изменение размеров, обе фигуры меняются

   if(Boublik *r = dynamic_cast<Boublik*>(figure1)) r-
>setsizes(70, 200);

   if(Boublik *r = dynamic_cast<Boublik*>(figure2)) r-
>setsizes(90, 120);

   getch();

   // проверяем перекраску, первая фигура не должна измениться

   if(FillBoublik *r = dynamic_cast<FillBoublik*>(figure1)) r-
>setfillcolor(RED);

   if(FillBoublik *r = dynamic_cast<FillBoublik*>(figure2)) r-
>setfillcolor(LIGHTRED);

   figure1->show();

   figure2->show();

   getch();

   //проверяем исчезновение с экрана при удалении

   delete figure1;

   delete figure2;

   getch();

   return 0;

}
```

4 Листинг реализации класса

```
void Figure::setcolor(int col){
```

```
   c = col;
   if(visible) draw();
}

void Figure::move(int x2, int y2){
   bool vis = visible;
   if(vis) hide();
   x = x2;
   y = y2;
   if(vis) show();
}

void Figure::hide(){
   if(!visible) 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) return;
   visible = 1;
   draw();
}

void Boublik::setsizes(int r_1, int r_2){
   bool vis = isvisible();
   if(vis) hide();
   r1 = r_1;
```

```
    r2 = r_2;
    if(vis) show();
}

void Boublik::area(int &x1, int &y1, int &x2, int &y2) const{
    x1 = x - r2;
    y1 = y - r2;
    x2 = x + r2;
    y2 = y + r2;
}

void Boublik::draw(){
    setcolor(getcolor());
    circle(x, y, r2);
    circle(x, y, r1);
}

void FillBoublik::setfillcolor(int fillcol){
    fc = fillcol;
    if(isvisible()) draw();
}

void FillBoublik::draw(){
    setfillstyle(SOLID_FILL, fc);
    fillellipse(x, y, r2, r2);
    setfillstyle(SOLID_FILL, BLACK);
    fillellipse(x, y, r1, r1);
}
```