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-211

_____ А.А.Владимиров

_____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

2. Эллипс
Ellipse(цвет линий, x и y центра, 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);

2 Описание интерфейса классов

```
   class Figure
   {
      private:
         int c; // цвет
         bool visible; // видимость
      protected:
         int x,y; // базовая точка
         virtual void draw(); // нарисовать
      public:
         Figure(int c, int x, int y): c(c), x(x),
y(y){visible = false;} // Конструктор
         virtual ~Figure() {} // Деструктор
         void move(int x, int y); // переместить фигуру в
точку
         void setcolor(int c); // установить цвет фигуры
         int getcolor() const { return c; } // получить цвет
         void hide(); // спрятать фигуру
         void show(); // показать фигуру
         bool isvisible() const { return visible; } //
видима?
         virtual void area(int &x1, int &y1, int &x2, int
&y2) const = 0;
      // получить размеры прямоугольной области, содержащей
фигуру
   };

   class Ellips: public Figure
   {
      protected:
         int r1, r2; // радиусы элипса
         void draw(); // нарисовать
      public:
         Ellips(int c, int x, int y, int r1, int r2):
Figure(c, x, y), r1(r1), r2(r2){} // конструктор
         ~Ellips(){hide();} // деструктор
         void setsizes(int r1, int r2);// изменение размера
         void area(int &x1,int &y1,int &x2,int &y2) const; //
получить размеры прямоугольной области, содержащей фигуру
   };

   class FillEllipse: public Ellips
   {
```

```
      private:
         int fillColor; // цвет закраски
         void draw(); // нарисовать
      public:
         FillEllipse(int c, int x, int y, int r1, int r2, int
fillColor): Ellips(c, x, y, r1, r2), fillColor(fillColor){}
//конструктор
         void setfillcolor(int c); // изменить цвет закраски
   };
```

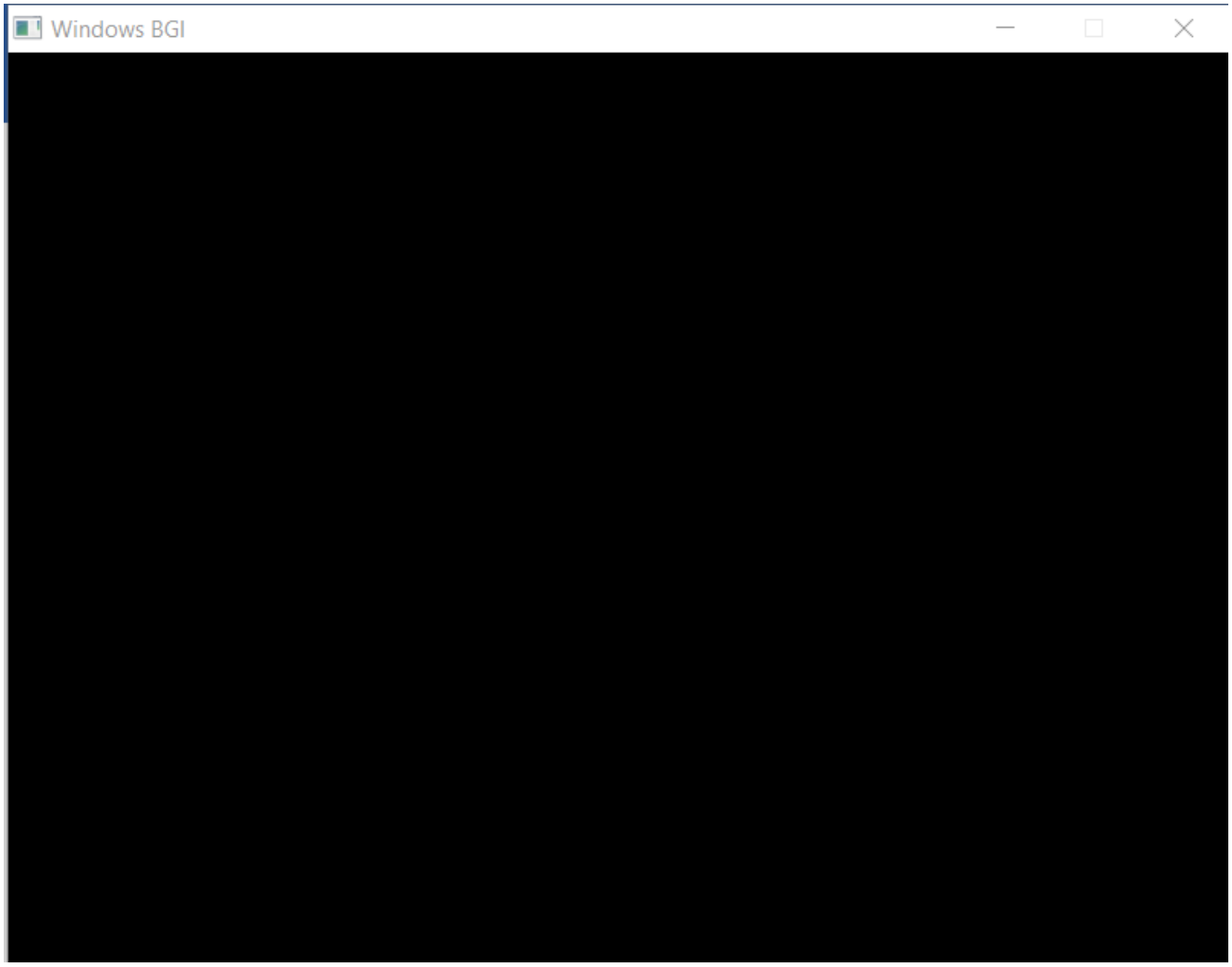
3 Описание тестов для проверки классов

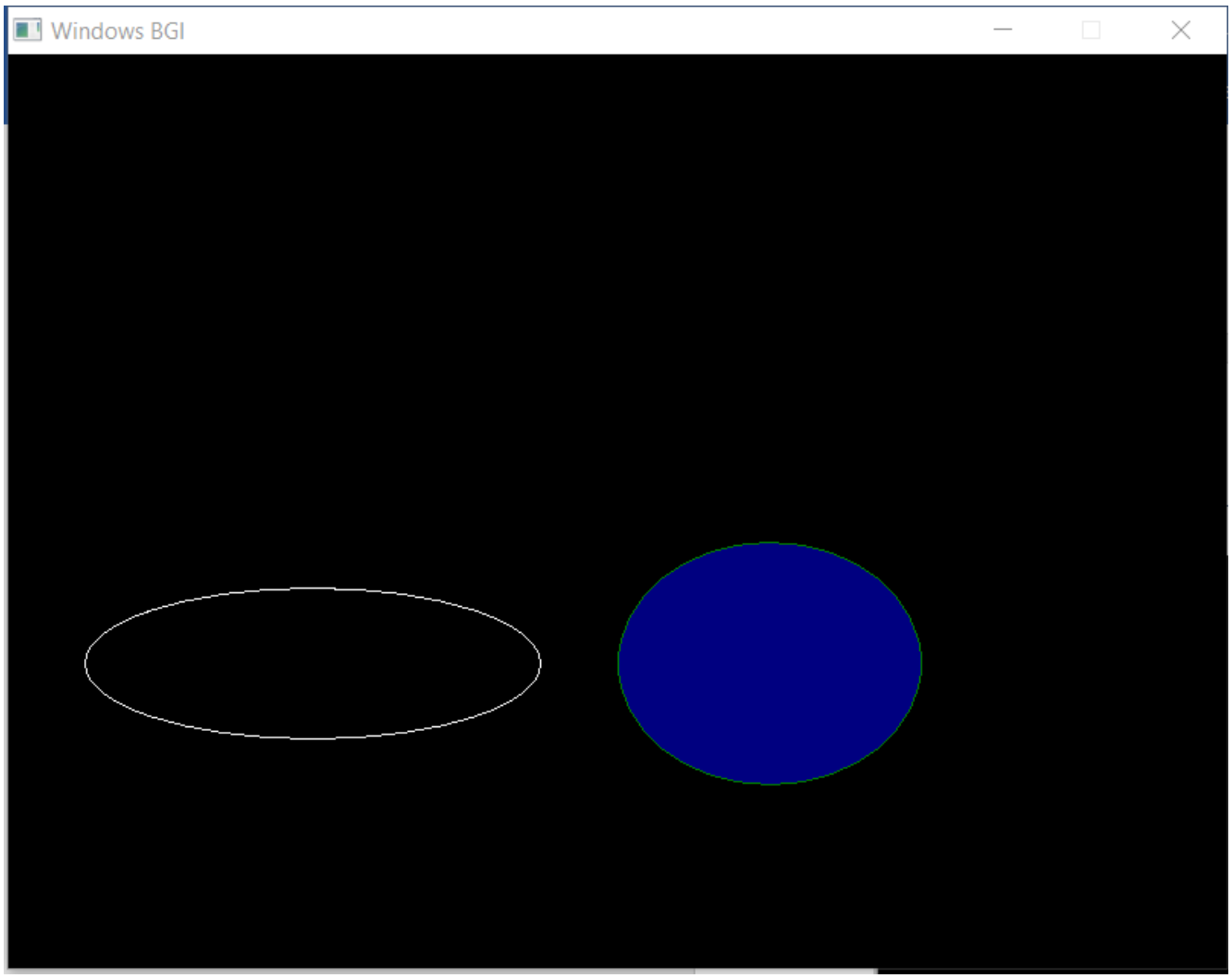
```
   int main() {
      initwindow(800, 600);
      Figure *a = new Ellips(GREEN, 200, 200, 150, 50);
      Figure *b = new FillEllipse(WHITE, 500, 200, 100, 80,
BLUE);

      a->show();
      b->show();
      getch();
      a->hide();
      b->hide();
      getch();
      a->move(200, 400);
      b->move(500, 400);
      a->show();
      b->show();
      getch();
      a->setcolor(WHITE);
      b->setcolor(GREEN);
      getch();
   // проверяем изменение размеров, обе фигуры меняются
      if (Ellips *r=dynamic_cast<Ellips *>(a)) r-
>setsizes(30, 200);
      if (Ellips *r=dynamic_cast<Ellips *>(b)) r-
>setsizes(100, 100);
      getch();
   // проверяем перекраску, фигура a не должна измениться
      if (FillEllipse *r=dynamic_cast<FillEllipse *>(a)) r-
>setfillcolor(YELLOW);
      if (FillEllipse *r=dynamic_cast<FillEllipse *>(b)) r-
>setfillcolor(YELLOW);
      getch();
   // проверяем исчезновение с экрана при удалении
      delete a;
      delete b;
```

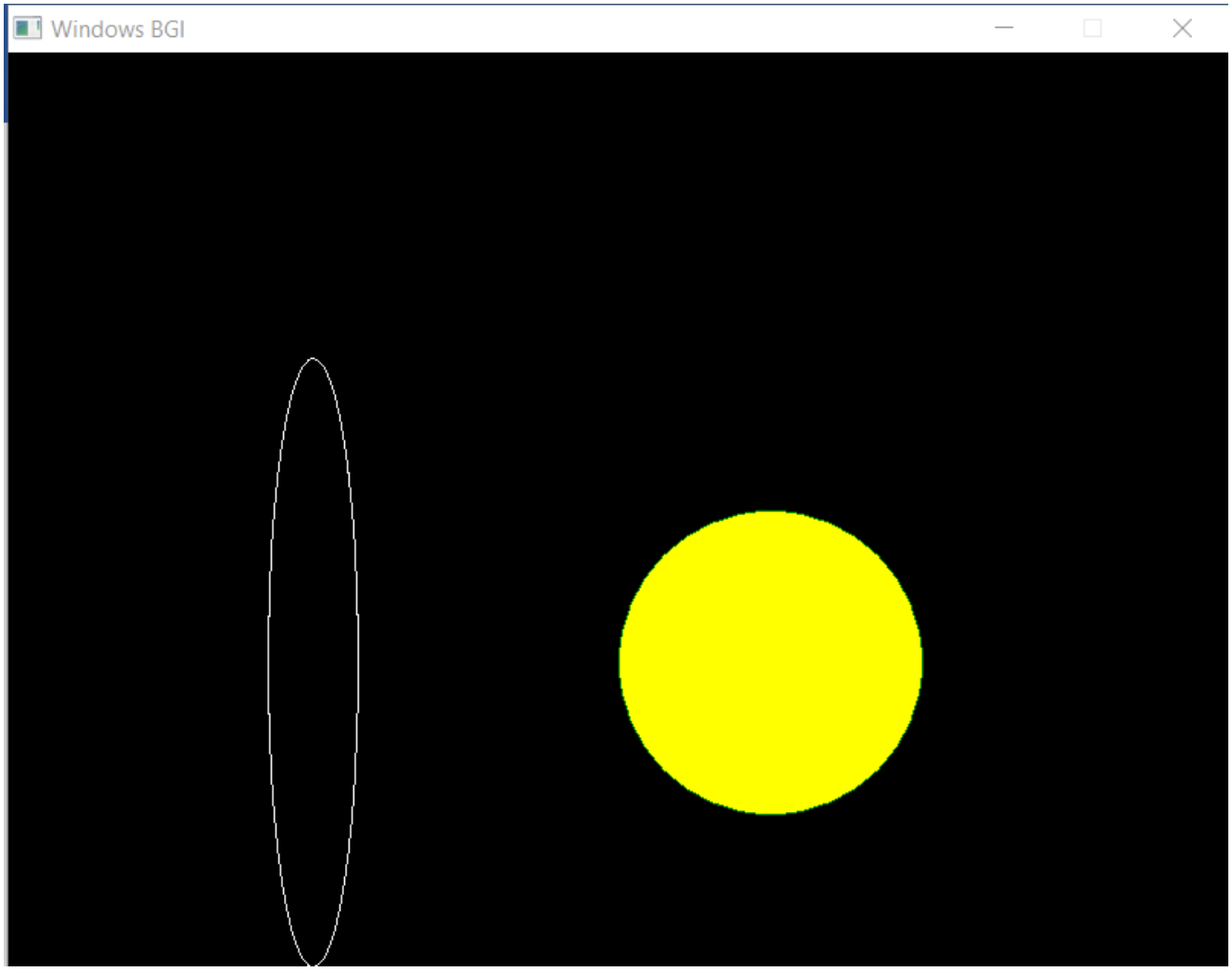
```
    getch();
    return 0;
}
```

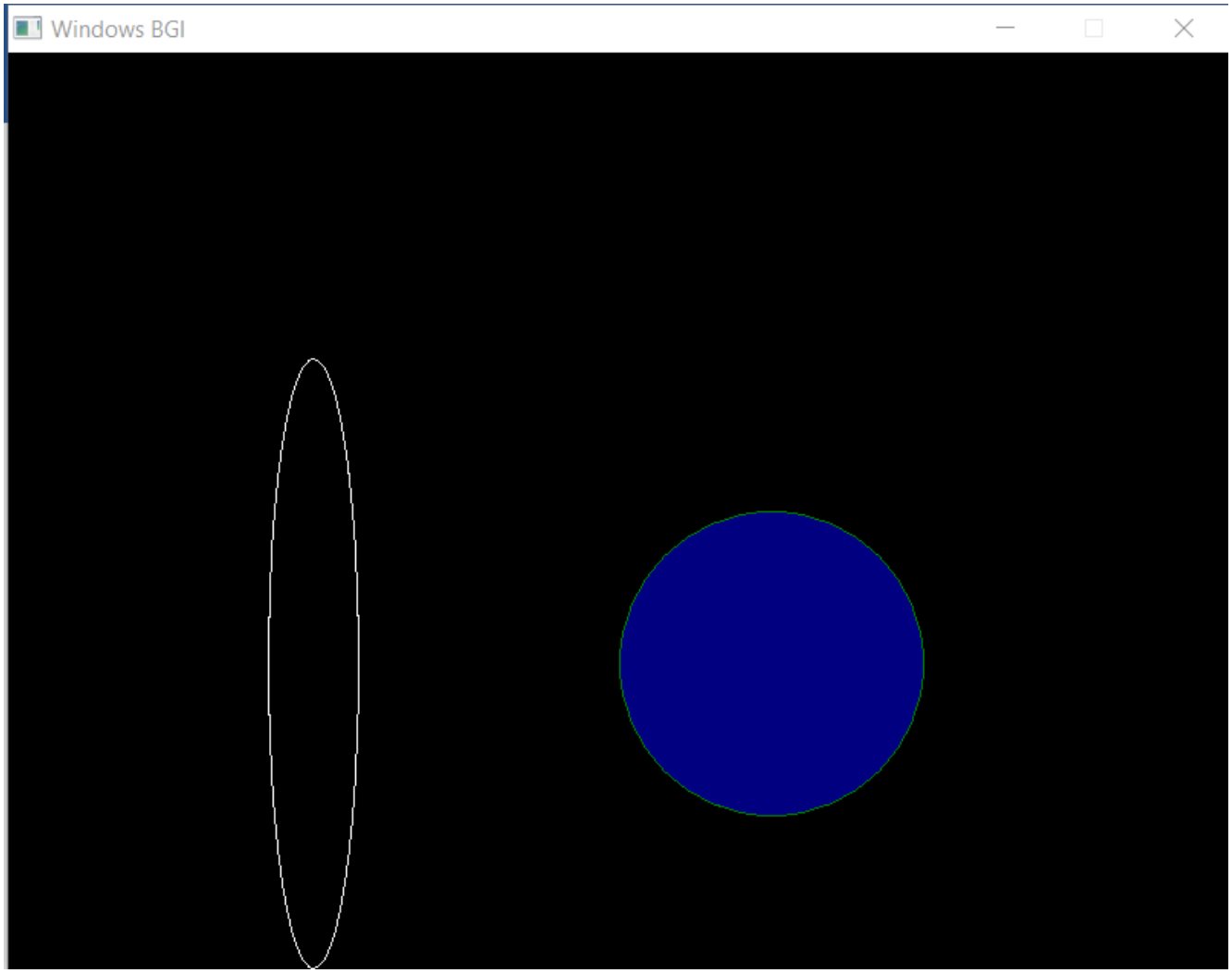
Полученные результаты


Windows BGI


Windows BGI


Windows BGI


Windows BGI



4 Листинг реализации класса

```
void Figure::setcolor(int c) {
   this->c = c;
   if (visible) draw();
}

void Figure::move(int x, int y) {
   bool s = visible;
   if (s) hide();
   this->x = x;
   this->y = y;
   if (s) 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) return;
   draw();
   visible = 1;
}

void Ellips::area(int &x1,int &y1,int &x2,int &y2) const {
   x1 = x-r1;
   y1 = y-r2;
   x2 = x+r1;
   y2 = y+r2;
}

void Ellips::setsizes(int r1, int r2) {
   bool s = isvisible();
   if (s) hide();
   this->r1 = r1;
   this->r2 = r2;
   if (s) show();
}

void Ellips::draw() {
   ::setcolor(getcolor());
   ellipse(x, y, 0, 360, r1, r2);
}

void FillEllipse::draw() {
   setfillstyle(SOLID_FILL, fillColor);
   Ellips::draw();
   floodfill(x, y, getcolor());
}

void FillEllipse::setfillcolor(int c) {
   fillColor = c;
   if (isvisible()) draw();
}
```