



**«Московский государственный технический университет
имени Н.Э. Баумана»
(национальный исследовательский университет)**

ФАКУЛЬТЕТ
КАФЕДРА

ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ
КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

О т ч е т

по домашнему заданию №3(часть 2)
варианта №7

Название лабораторной работы:

Наследование

Дисциплина:

Основы программирования

Студент гр. **ИУ6-12**

30/12/17 _____
(Подпись, дата)

Векшин Роман
(И.О. Фамилия)

Преподаватель

(Подпись, дата)

Черноусова Татьяна Геннадьевна
(И.О. Фамилия)

Задание

Разработать и реализовать иерархию классов для описанных объектов предметной области, используя механизмы наследования. Проверить ее на тестовом примере, с демонстрацией всех возможностей разработанных классов на конкретных данных.

Объект – Треугольник, заданный точками на плоскости. Объект умеет инициализировать поля, выводить на экран значение своих полей, отвечать на запрос об этих значениях и вычислять площадь фигуры.

Объект – Треугольная призма. Объект умеет инициализировать поля, выводить на экран содержимое своих полей, возвращать по запросу их значения и площадь развертки.

Диаграмма классов

Таблица 1-Диаграмма класса Треугольник

triangle
x1, x2, x3, y1, y2, y3
get_square() get_perimeter() get_x1() get_x2() get_x3() get_y1() get_y2() get_y3() display_field() Init()

Таблица 2-Диаграмма класса Призма

prism
h
get_square() get_h() display_field() Init()

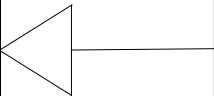


Схема алгоритма

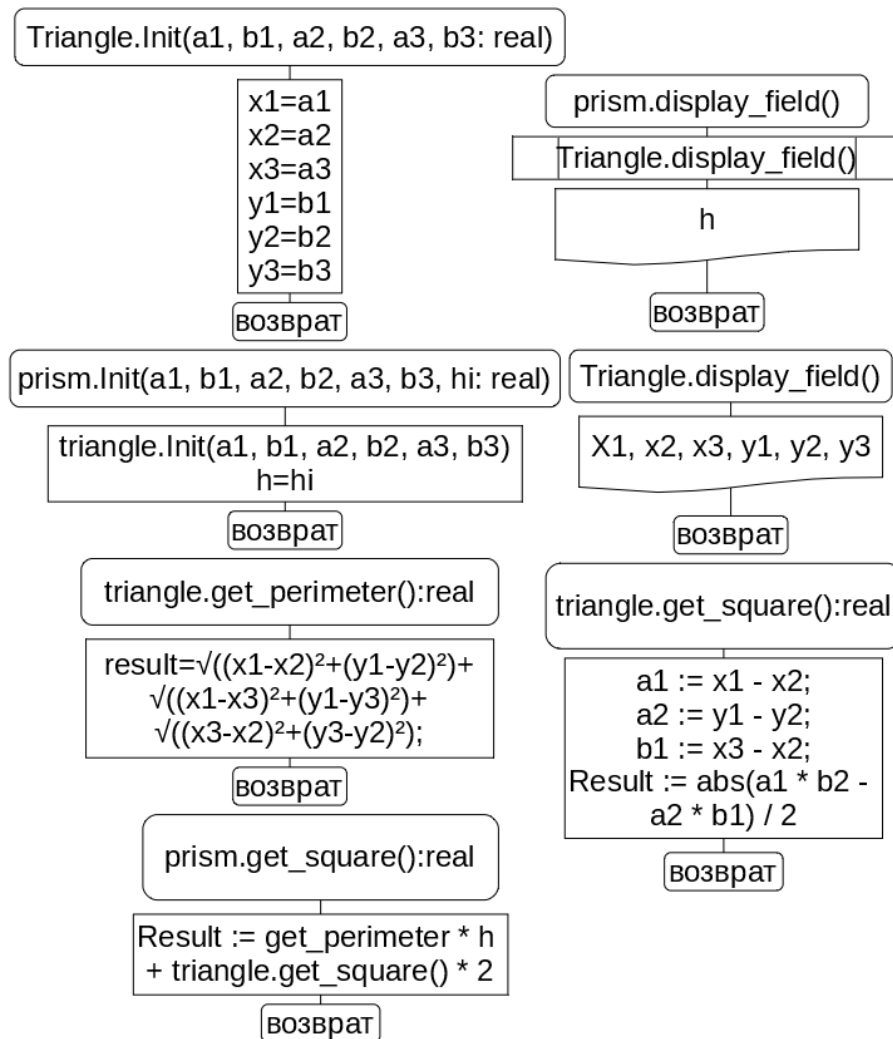


Рисунок 1-Схема алгоритма

Код программы

```

program project1;
{$APPTYPE CONSOLE}

type
  triangle = object
    x1, x2, x3, y1, y2, y3: real;
    procedure Init(a1, b1, a2, b2, a3, b3: real);
    function get_square(): real;
    function get_perimeter(): real;
    function get_x1(): real;
    function get_x2(): real;
    function get_x3(): real;
    function get_y1(): real;
    function get_y2(): real;
    function get_y3(): real;
    procedure display_field();
  end;

  prism = object(triangle)
    h: real;
    procedure Init(a1, b1, a2, b2, a3, b3, hi: real);
    function get_h(): real;
    function get_square(): real;
    procedure display_field();
  end;

  procedure triangle.Init(a1, b1, a2, b2, a3, b3: real);

```

```

begin
  x1 := a1;
  x2 := a2;
  x3 := a3;
  y1 := b1;
  y2 := b2;
  y3 := b3;
end;

function triangle.get_x1(): real;
begin
  Result := x1;
end;

function triangle.get_x2(): real;
begin
  Result := x2;
end;

function triangle.get_x3(): real;
begin
  Result := x3;
end;

function triangle.get_y1(): real;
begin
  Result := y1;
end;

function triangle.get_y2(): real;
begin
  Result := y2;
end;

function triangle.get_y3(): real;
begin
  Result := y3;
end;

procedure triangle.display_field();
begin
  writeln('x1: ', x1: 3: 6);
  writeln('x2: ', x2: 3: 6);
  writeln('x3: ', x3: 3: 6);
  writeln('y1: ', y1: 3: 6);
  writeln('y2: ', y2: 3: 6);
  writeln('y3: ', y3: 3: 6);
end;

procedure prism.display_field();
begin
  triangle.display_field();
  writeln(' h: ', h: 3: 6);
end;

procedure prism.Init(a1, b1, a2, b2, a3, b3, hi: real);
begin
  triangle.Init(a1, b1, a2, b2, a3, b3);
  h := hi;
end;

function triangle.get_perimeter(): real;
begin
  Result := sqrt(sqr(x1 - x2) + sqr(y1 - y2)) + sqrt(sqr(x1 - x3) + sqr(y1 - y3)) +
    sqrt(sqr(x3 - x2) + sqr(y3 - y2));
end;

function triangle.get_square(): real;
var
  a1, a2, b1, b2: real;
begin
  a1 := x1 - x2;
  a2 := y1 - y2;
  b1 := x3 - x2;

```

```

    b2 := y3 - y2;
    Result := abs(a1 * b2 - a2 * b1) / 2;
end;

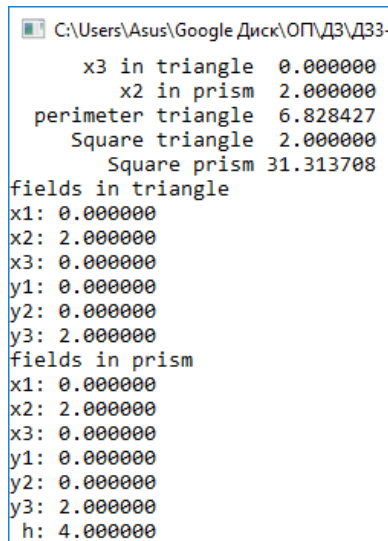
function prism.get_square(): real;
begin
    Result := get_perimeter * h + triangle.get_square() * 2;
end;

function prism.get_h(): real;
begin
    Result := h;
end;

var
    t: triangle;
    q: prism;
begin
    t.Init(1, 1, -2, 4, -2, -2);
    q.Init(1, 1, -2, 4, -2, -2, 4);
    writeln('x3 in triangle': 20, t.get_x3(): 10: 6);
    writeln('x2 in prism': 20, q.get_x2(): 10: 6);
    writeln('perimeter triangle': 20, t.get_perimeter(): 10: 6);
    writeln('Square triangle': 20, t.get_square(): 10: 6);
    writeln('Square prism': 20, q.get_square(): 10: 6);
    writeln('fields in triangle');
    t.display_field();
    writeln('fields in prism');
    q.display_field();
    readln();
end.

```

Пример работы программы

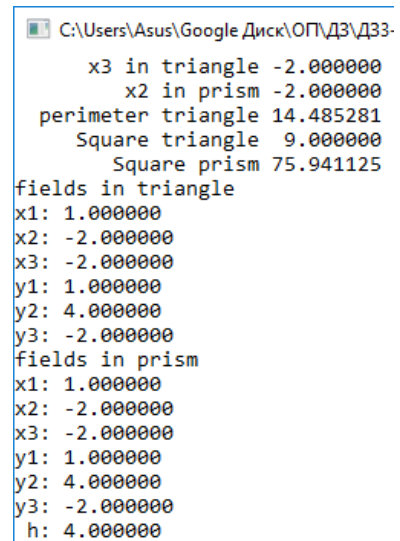


```

C:\Users\Asus\Google Диск\ОП\ДЗ\ДЗ3-
    x3 in triangle  0.000000
    x2 in prism    2.000000
    perimeter triangle  6.828427
    Square triangle  2.000000
    Square prism    31.313708
    fields in triangle
    x1: 0.000000
    x2: 2.000000
    x3: 0.000000
    y1: 0.000000
    y2: 0.000000
    y3: 2.000000
    fields in prism
    x1: 0.000000
    x2: 2.000000
    x3: 0.000000
    y1: 0.000000
    y2: 0.000000
    y3: 2.000000
    h: 4.000000

```

Рисунок 2-Пример работы программы



```

C:\Users\Asus\Google Диск\ОП\ДЗ\ДЗ3-
    x3 in triangle -2.000000
    x2 in prism    -2.000000
    perimeter triangle  14.485281
    Square triangle  9.000000
    Square prism    75.941125
    fields in triangle
    x1: 1.000000
    x2: -2.000000
    x3: -2.000000
    y1: 1.000000
    y2: 4.000000
    y3: -2.000000
    fields in prism
    x1: 1.000000
    x2: -2.000000
    x3: -2.000000
    y1: 1.000000
    y2: 4.000000
    y3: -2.000000
    h: 4.000000

```

Рисунок 3-Пример работы программы

Вывод

- 1) Разработаны модели объектов Треугольник и Треугольная призма (см. таблицы 1-2) и алгоритм тестирующей программы, и составлены их схемы в среде LibreOffice Draw (см. рис. 1).
- 2) Создан код программы по схемам алгоритма и моделей объектов в среде Lazarus.
- 3) Проведено тестирование.
- 4) Тестирование показало корректность работы программы (см. рис. 2-3).