Контрольная работа по ОБЪЕКТНОМУ ПРОГРАММИРОВАНИЮ

2-й курс, 2013 спецкурс по выбору Плещинский Н.Б.

Для положительного результата достаточно решить любые две задачи!

1. Написать программу для решения в комплексных числах квадратного уравнения (на любом языке программирования).
   Если на Турбо Паскале, то можно использовать библиотечный модуль Complex, интерфейсная часть прилагается ниже.

2. Описать класс (тип объекта) Triangle (треугольник), полями которого являются координаты вершин. Предусмотреть виртуальный метод, возвращающий длины сторон треугольника.
   Написать процедуру, позволяющую узнать, равны или нет два треугольника.

3. Написать на Турбо Паскале программу, перемещающую по экрану монитора крестик ┼ (клавишами со стрелками). Использовать библиотечный модуль Figures, текст упрощенного варианта прилагается.
   { Процедура Line(x1,y1,x2,y2) из модуля Graph рисует отрезок, соединяющий точки с указанными координатами }

4. Описать (на любом языке) класс «интегральное уравнение Вольтера 2-го рода». Предусмотреть метод для нахождения численного решения методом механических квадратур.

5. Трудная задача, не обязательная!
   Написать на абстрактном уровне программу, реализующую процесс ортогонализации Шмидта – по системе линейно независимых элементов строится система ортогональных элементов. Какие будут элементы – станет известно когда-нибудь потом.

ПРИЛОЖЕНИЕ

(* Интерфейсная часть модуля Complex (к задаче 1) *)

```
UNIT Complex;
INTERFACE
type
 CompP = ^Comp;
 Comp = object
  Re,Im: real;
  procedure Zero;
  procedure InpRI(x,y: real);
  function  OutRe: real;
  function  OutIm: real;
  procedure Add(z: Comp);
  procedure AddRI(x,y: real);
  procedure Sub(z: Comp);
  procedure SubRI(x,y: real);
  procedure Mul(z: Comp);
  procedure MulRI(x,y: real);
  procedure Dvd(z: Comp);
  procedure DvdRI(x,y: real);
  function  Modul: real;
  function  Argum: real;
  end;
```

```
(* Модуль : АБСТРАКТНЫЕ ФИГУРЫ   (к задаче 3)   *)
UNIT Figures;
INTERFACE
uses CRT;
type
  Figure = object
    X,Y : word;
    Vis: boolean;
    constructor Init(X_,Y_: word);
    destructor Done;
    procedure Show; virtual;
    procedure Hide; virtual;
    procedure MoveTo(X_,Y_: integer);
    procedure Drag(Step: integer); virtual;
    end;
IMPLEMENTATION
constructor Figure.Init;
  begin  X:=X_; Y:=Y_; Vis:=false  end;
destructor Figure.Done;
  begin  Hide  end;
procedure Figure.Show;
  begin  Vis:=true  end;
procedure Figure.Hide;
  begin  Vis:=false  end;
procedure Figure.MoveTo;
  begin
  if Vis
  then begin Hide; X:=X_; Y:=Y_; Show  end
  else begin X:=X_; Y:=Y_  end
  end;
function Delta(var DX,DY: integer): boolean;  {возвращает приращения}
  var Key: char; Press: boolean;              {координат и False, если}
  begin                                       {нажата клавиша Enter}
  DX:=0; DY:=0; Delta:=True;
  repeat
    Key:=ReadKey; Press:=True;   {ReadKey из CRT возвращает КОДЫ}
    case Ord(Key) of             {нажатой клавиши}
      0: begin
         Key:=ReadKey;
         case Ord(Key) of
           72: DY:=-1;           {клавиши со стрелками генерируют}
           80: DY:=1;            {два кода: 0 и 72,80,75 или 77}
           75: DX:=-1;
           77: DX:=1;
           else Press:=False;
           end
         end;
      13: Delta:=False;          {а клавиша Enter - только код 13}
      else Press:=False
      end
    until Press
  end;

procedure Figure.Drag;  {фигура перемещается по экрану, если нажимать}
  var DX,DY: integer;   {клавиши со стрелками. Enter - выход}
  begin
  while Delta(DX,DY) do MoveTo(X+DX*Step,Y+DY*Step)
  end;
END.
```