Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Воронежский государственный технический университет»

Кафедра конструирования и производства радиоаппаратуры

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению лабораторных работ № 3-4 по дисциплине «Информационные технологии» для студентов направления 11.03.03 «Конструирование и технология электронных средств» (профиль «Проектирование и технология радиоэлектронных средств»)
всех форм обучения

Воронеж 2021

УДК 004.432.2
ББК 32.97

Составители:
д-р. техн. наук М.А. Ромащенко,
канд. техн. наук А.А. Пирогов,
И.В. Свиридова

Методические указания к выполнению лабораторных работ № 3-4 по дисциплине «Информационные технологии» для студентов направления 11.03.03 «Конструирование и технология электронных средств» (профиль «Проектирование и технология радиоэлектронных средств») всех форм обучения / ФГБОУ ВО «Воронежский государственный технический университет»; сост. М.А. Ромащенко, А.А. Пирогов, И.В. Свиридова. Воронеж: Изд-во ВГТУ, 2021, 20 с.

Методические указания предназначены для выполнения лабораторных работ № 3-4 по дисциплине «Информационные технологии» студентами очной и заочной форм обучения.

Предназначены для студентов первого курса обучения.

Методические указания подготовлены в электронном виде и содержатся в файле ЛР3-4.pdf.

Библиогр.: 4 назв.

УДК 004.432.2
ББК 32.97

Рецензент - О. Ю. Макаров, д-р техн. наук, проф.
кафедры конструирования и производства
радиоаппаратуры ВГТУ

Издается по решению редакционно-издательского совета Воронежского государственного технического университета

1. ЛАБОРАТОРНАЯ РАБОТА №3

ИЗУЧЕНИЕ ГРАФИЧЕСКИХ ВОЗМОЖНОСТЕЙ ЯЗЫКА ПАСКАЛЬ.

Цель работы: изучить назначение модуля Graph языка Паскаль. Закрепить знания об основных процедурах и функциях работы с графикой на языке Паскаль. Освоить методы изображения простейших геометрических фигур и вывода текста в графическом режиме.

Время работы: 8 часов.

1.1. Домашние задания и методические указания по их выполнению

Задание 1 – изучить назначение модуля Graph и графического режима языка Паскаль.

В состав Паскаля включена мощная библиотека графических подпрограмм Graph. Библиотека содержит в общей сложности более 50 процедур и функций, предоставляющих программисту самые разнообразные средства управления графическим экраном.

Стандартное состояние компьютера в момент запуска программы из среды ТурбоПаскаль соответствует текстовому режиму. Для использования графических средств необходимо определенным образом инициировать графический режим работы видеоадаптера. Это достигается подключением нужного графического драйвера. Обычно они располагаются в папке BGI с расширением BGI, например, EGAVGA.BGI

Процедура **InitGraph** инициирует графический режим работы адаптера. Заголовок процедуры имеет вид:

procedure InitGraph(**var** Driver, Mode: Integer; Path: String);

где, Driver определяет тип графического драйвера, а Mode

задает режим его работы; Path – указывает путь где расположен драйвер.

Если программа рассчитана на работу с любым адаптером, то обращение к процедуре выполняется с требованием автоматического определения типа драйвера

Driver:=Detect;

InitGraph(Driver, Mode, ``);

Path можно не указывать, при условии когда нужный драйвер находится в текущем каталоге.

Процедура **CloseGraph** завершает работу адаптера в графическом режиме и восстанавливает текстовый режим работы.

В графическом режиме указатель текущей позиции на экране, в отличии от текстового курсора, невидим. Положение и любая координата на графическом экране задается относительно левого верхнего угла, имеющего координаты 0,0.

Задание 2 – закрепить знания об основных процедурах и функциях работы с графикой на языке Паскаль.

Функции **GetMaxX** и **GetMaxY** возвращают значения содержащие максимальные координаты экрана в текущем режиме по горизонтали и вертикали.

Процедура **SetViewPort** устанавливает прямогольное окно на графическом экране. Заголовок процедуры имеет вид:

procedure SetViewPort(X1,Y1,X2,Y2: Integer; ClipOn: Boolean);

X1,Y1 – координаты левого верхнего угла окна, X2,Y2 – правого нижнего. ClipOn определяет отсечку не умещающихся в окне элементов изображения – True элементы отсекаются, False – границы окна игнорируются.

Процедура **MoveTo** устанавливает новое текущее положение указателя. Заголовок процедуры имеет вид:

procedure MoveTo(X,Y: Integer);

Процедура **MoveRel** устанавливает новое положение указателя в относительных единцах. Заголовок процедуры имеет вид:

procedure MoveRel(DX,DY: Integer);

Процедура **ClearDevice** очищает графический экран. После обращения к процедуре указатель устанавливается в левый верхний угол экрана.

Процедура **ClearViewPort** очищает графическое окно, а если оно не определено то весь экран. После обращения к процедуре указатель устанавливается в левый верхний угол экрана.

Процедура **GetAspectRatio** возвращает два числа, позволяющих оценить соотношение сторон экрана. Заголовок процедуры имеет вид:

procedure GetAspectRatio(**var** X,Y: Word);

Значения этих переменных позволяют вычислить отношение сторон графического экрана в пикселях и использовать его при построении правильных геометрических фигур.

Пример построение правильного квадрата со стороной L пикселов по вертикали

```
GetAspectRatio(Xasp,Yasp);
Rectangle(X1, Y1, X1+L*round(Yasp/Xasp), Y1+L);
```

Задание 3 – закрепить знания о процедурах и функциях рисования линий и точек на языке Паскаль.

Процедура **PutPixel** выводит заданным цветом точку по указанным координатам. Заголовок процедуры имеет вид:

procedure PutPixel(X,Y: Integer; Color: Word);

Процедура **Line** вычерчивает линию с указанными координатами начала и конца. Заголовок процедуры имеет вид:

procedure Line(X1,Y1,X2,Y2: Integer);

где, X1,Y1 – координаты начала линии, а X2,Y2 – конца.

Процедура **LineTo** вычерчивает линию от текущего положения указателя до положения, заданного новыми координатами. Заголовок процедуры имеет вид:

procedure LineTo(X,Y: Integer);

Процедура **LineRel** вычерчивает линию от текущего положения указателя до положения, заданного приращениями его координат. Заголовок процедуры имеет вид:

procedure LineRel(DX,DY: Integer);

Процедура **SetLineStyle** устанавливает новый стиль вычерчиваемых линий. Заголовок процедуры имеет вид:

procedure SetLineStyle(Type, Pattern, Thick: Word);

где, Type, Pattern, Thick – соответственно тип, образец и толщина вычерчиваемых линий. Тип линий может быть задан с помощью одной из следующих констант

SolidLn = 0 {сплошная}
DottedLn = 1 {точечная}
CenterLn = 2 {штрих-пунктирная}
DashedLn = 3 {пунктирная}

Установленный процедурой стиль линий используется при построении прямоугольников и других фигур.

Процедура **SetWriteMode** устанавливает способ взаимодействия вновь выводимых линий с уже существующим на экране изображением. Заголовок процедуры имеет вид:

procedure SetWriteMode(Mode);

Если параметр Mode имеет значение 0, выводимые линии накладываются на существующее изображение обычным образом, если значение равно 1, то это наложение осуществляется через «логическое ИЛИ», т.е. в точках пересечения светимость пикселей инвертируется.

Задание 4 – закрепить знания о процедурах и функциях рисования прямоугольников и окружностей на языке Паскаль.

Процедура **Rectangle** вычерчивает прямоугольник с указанными координатами углов. Заголовок процедуры имеет вид:

procedure Rectangle(X1,Y1,X2,Y2: Integer);

где, X1-Y2 координаты левого верхнего (X1,Y1) и правого нижнего (X2,Y2) углов прямоугольника. Он вычерчивается с использованием текущих цвета и стиля линий.

Процедура **DrawPoly** вычерчивает произвольную ломаную линию, заданную координатами точек излома. Заголовок процедуры имеет вид:

procedure DrawPoly(N: Word; **var** Points);

где, N - количество точек излома, включая обе крайние точки; Points - переменная типа PointType, содержащая координаты точек излома. Точки излома задаются парой значений типа Word: первое определяет горизонтальную, второе - вертикальную координаты. Для них используется специальный определенный в модулей тип PointType. При вычерчивании используется текущие цвет и стиль линий.

Процедура **Circle** вычерчивает окружность. Заголовок процедуры имеет вид:

procedure Circle(X,Y: Integer; R: Word);

где, X, Y – координаты центра; R – радиус в пикселях. Окружность выводится текущим цветом. Толщина линии устанавливается текущим стилем, но вид линии всегда SolidLn (сплошная).

Процедура **Arc** чертит дугу окружности. Заголовок процедуры имеет вид:

procedure Arc(X,Y: Integer; BegA,EndA,R: Word);

где, X, Y – координаты центра; BegA, EndA – соответственно начальный и конечный углы дуги; R – радиус. Углы отсчитываются против часовой стрелки и указываются в градусах. Нулевой угол соответствует горизонтальному направлению вектора слева направо.

Процедура **Ellipse** чертит дугу эллипса. Заголовок процедуры имеет вид:

procedure Ellipse(X,Y: Integer; BegA,EndA,RX,RY: Word);

где, X, Y – координаты центра; BegA, EndA – соответственно начальный и конечный углы дуги; RX, RY – горизонтальный и вертикальный радиусы эллипса в пикселях.

Задание 5 – закрепить знания о процедурах и функциях вывода текста в графическом режиме на языке Паскаль.

Процедура **OutText** выводит текстовую строку, начиная с текущего положения указателя. Заголовок процедуры имеет вид:

procedure OutText(Txt: String);

Процедура **OutTextXY** выводит текстовую строку, начиная с заданного места. Заголовок процедуры имеет вид:

procedure OutTextXY(X, Y: Integer; Txt: String);

Процедура **SetTextStyle** устанавливает стиль текстового вывода на графический экран. Заголовок процедуры имеет вид:

procedure SetTextStyle(Font, Direct, Size: Word);

где, Font – код шрифта, Direct – код направления (0 – слева направо, 1 – снизу вверх), Size – код размера шрифта.

Процедура **SetTextJustify** задает выравнивание выводимого текста по отношению к текущему положению указателя или к заданным координатам. Заголовок процедуры имеет вид:

procedure SetTextJustify(Horiz, Vert: Word);

где, Horiz – горизонтальное выравнивание, Vert – вертикальное.

2.2. Лабораторные задания

Задание 1 – Наберите следующий программный код. Проанализируйте назначение каждого оператора, попробуйте самостоятельно поменять какие-либо параметры и посмотреть какие это вызовет изменения.

```
program Gr;
uses crt, graph;
var
//Переменные для графического драйвера
    grDriver:Integer; //Графический драйвер
    grMode:Integer;   //Режим графического драйвера
    i,j:Integer;
    st:String;
begin
    grDriver:=Detect;
    InitGraph(grDriver,grMode,''); //Инициализация модуля Graph
    OutTextXY(90,340,'SetColor()'); //Вывод текста
    for i:=1 to 16 do
     begin
```

```
        SetColor(i); //Цвет лини
        Line(i*20,1,i*20,300); //Вывод линии
        str(i,st);
        OutTextXY(i*20,360,st);
      end;
    ReadKey;
    ClearDevice; //Очистка эрана
    SetColor(15);
    OutTextXY(90,340,'SetLineStyle(i,1,1)');
    for i:=1 to 16 do
      begin
        SetLineStyle(i,1,1); //Изменение стиля линии
        Line(i*20,1,i*20,300);
        str(i,st);
        OutTextXY(i*20,360,st);
      end;
    ReadKey;
    ClearDevice;
    SetColor(15);
    OutTextXY(90,340,'SetLineStyle(1,1,i)'); //Изменение ширины
линии
    for i:=1 to 16 do
      begin
        SetLineStyle(1,1,i);
        Line(i*20,1,i*20,300);
        str(i,st);
        OutTextXY(i*20,360,st);
      end;
    ReadKey;
    CloseGraph; //Выход из графического режима
  end.
```

Задание 2 – Составьте блок-схему алгоритма и напишите программу на языке Паскаль, которая создаст таблицу с характеристиками в соответствии с выданным вариантом. Затем

заполните двухмерный массив случайными целыми числами и результат выведите в нарисованную таблицу.

1. таблица 5x5, горизонтальные линии красные, вертикальные белые
2. таблица 3x8, горизонтальные линии желтые, вертикальные малиновые
3. таблица 4x7, горизонтальные линии зеленые, вертикальные розовые
4. таблица 5x6, горизонтальные линии синие, вертикальные коричневые
5. таблица 6x3, горизонтальные линии бирюзовые, вертикальные фиолетовые
6. таблица 7x5, горизонтальные линии фиолетовые, вертикальные бирюзовые
7. таблица 4x6, горизонтальные линии коричневые, вертикальные синие
8. таблица 3x7, горизонтальные линии розовые, вертикальные зеленые
9. таблица 7x7, горизонтальные линии малиновые, вертикальные желтые
10. таблица 5x7, горизонтальные линии белые, вертикальные красные

Задание 2 – Составьте блок-схему алгоритма и напишите программу на языке Паскаль, которая создаст заданное количество окружностей с увеличивающимся радиусом. Окружности должны быть разного цвета. Характеристики и число окружностей взять в соответствии с выданным вариантом

1. число окружностей 5, центр с координатами 300x240, шаг увеличения радиуса 25
2. число окружностей 6, центр с координатами 290x250, шаг увеличения радиуса 20
3. число окружностей 7, центр с координатами 280x260, шаг

увеличения радиуса 15

4. число окружностей 8, центр с координатами 270x270, шаг увеличения радиуса 10

5. число окружностей 9, центр с координатами 260x280, шаг увеличения радиуса 5

6. число окружностей 4, центр с координатами 200x200, шаг увеличения радиуса 30

7. число окружностей 5, центр с координатами 100x100, шаг увеличения радиуса 10

8. число окружностей 6, центр с координатами 120x120, шаг увеличения радиуса 15

9. число окружностей 7, центр с координатами 140x140, шаг увеличения радиуса 20

10. число окружностей 8, центр с координатами 160x160, шаг увеличения радиуса 10

2.3. Контрольные вопросы для отчета работы

1. Для чего необходим модуль Graph языка Паскаль?
2. Что такое графический режим? Как происходит инициация графического режима?
3. Что такое графический драйвер? Что такое пиксел?
4. Где находится начало отсчета графических координат?
5. Перечислите основные процедуры и функции для установки положения курсора в графическом режиме языка Паскаль.
6. Перечислите основные процедуры и функции для рисования линий и точек в графическом режиме языка Паскаль.
7. Перечислите основные процедуры и функции для рисования прямоугольников и окружностей в графическом режиме языка Паскаль.
8. Перечислите основные процедуры и функции для вывода текста в графическом режиме языка Паскаль.

2. ЛАБОРАТОРНАЯ РАБОТА №4

ПОСТРОЕНИЕ ГРАФИКОВ ФУНКЦИЙ И СОЗДАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА НА ЯЗЫКЕ PASCAL С ПОМОЩЬЮ МОДУЛЯ GRAPH.

Цель работы: расширить знания об использовании процедур и функций модуля Graph языка Паскаль. Научиться строить графики функции и круговые диаграммы. Получить навык создания графического интерфейса средствами языка Паскаль.

Время работы: 8 часов.

2.1. Домашние задания и методические указания по их выполнению

Задание 1 – закрепить знания о процедурах и функциях для работы с красками, палитрами и заполнениями в графическом режиме на языке Паскаль.

Процедура **SetColor** устанавливает текущий цвет для выводимых линий и символов. Заголовок процедуры имеет вид:

procedure SetColor(Color: Word);

Процедура **SetBkColor** устанавливает цвет фона. Заголовок процедуры имеет вид:

procedure SetBkColor(Color: Word);

Процедура **SetFillStyle** устанавливает тип и цвет заполнения. Заголовок процедуры имеет вид:

procedure SetFillStyle(Fill, Color: Word);

За счет заполнения можно покрывать какие-либо фрагменты изображения периодически повторяющимся узором. Для указания типа заполнения используются следующие, предварительно определенные константы

```
SolidFill       = 1;  {сплошное заполнение}
LineFill        = 2;  {заполнение символами --------}
```

LtSlashFill = 3; {заполнение символами ////////}
SlashFill = 4; {заполнение утолщенными символами ////////}
BkSlashFill = 5; {заполнение утолщенными символами \\\\\\\\}
LtBkSlashFill = 6; {заполнение символами \\\\\\\}
HatchFill = 7; {заполнение символами ++++++}
XhatchFill = 8; {заполнение символами xxxxxxx}
IntegerLeaveFill= 9; {заполнение символами #######}
WideDotFill = 10; {заполнение редкими точками}
CloseDotFill = 11; {заполнение частыми точками}

Процедура **FloodFill** заполняет произвольную замкнутую фигуру используя текущий стиль заполнения. Заголовок процедуры имеет вид:

procedure FloodFill(X, Y: Integer; Border: Word);

где, X, Y – координаты любой точки внутри замкнутой фигуры; Border – цвет граничной линии.

Процедура **Bar** заполняет прямоугольную область экрана текущим стилем. Заголовок процедуры имеет вид:

procedure Bar(X1, Y1, X2, Y2, Depth: Integer);

Процедура **Bar3D** вычерчивает трехмерное изображение параллелепипеда и закрашивает его переднюю грань. Заголовок процедуры имеет вид:

procedure Bar3D(X1, Y1, X2, Y2: Integer; Top: Boolean);

где, Top – способ изображения верхней грани (True – верхняя грань вычерчивается). Передняя грань заливается текущим стилем заполнения.

Процедура **FillPoly** обводит линией и закрашивает замкнутый многоугольник. Заголовок процедуры имеет вид:

procedure FillPoly(N: Word; **var** Coords);

где, N – количество вершин замкнутого многоугольника; Coord – переменная типа PoinType, содержащая координаты вершин.

Для построения круговых диаграмм удобно использовать процедуру PieSlice, которая вычерчивает и заполняет сектор окружности. Заголовок процедуры имеет вид:

procedure PieSlice(X, Y: Integer; BegA, EndA, R: Word);

где, BegA и EndA начальный и конечный угол сектора

Задание 2 – закрепить знания о процедурах и функциях для сохранения и вывода изображений в графическом режиме на языке Паскаль.

Процедура **GetImage** помещает в память копию прямоугольного фрагмента изображения. Заголовок процедуры имеет вид:

procedure GetImage(X1, Y1, X2, Y2: Integer; **var** Buf);

где, Buf – переменная в которую будет помещена копия видеопамяти с фрагментом изображения.

Процедура **PutImage** выводит в заданное место экрана копию фрагмента изображения. Заголовок процедуры имеет вид:

procedure PutImage(X, Y: Integer; **var** Buf; Mode: Word);

где, Mode – определяет способ взаимодействия размещаемой копии и существующего изображения.

NormalPut= 0; {замена существующего изображения на копию}
XORPut = 1; {исключающее ИЛИ}
ORPut = 2; {объединяющее ИЛИ}
ANDPut = 3; {логическое И}
NOTPut = 4; {инверсия изображения}

Задание 3 – изучить основные принципы создания графического интерфейса на языке Паскаль.

Ниже приведен пример построения графического интерфейса с помощью модуля Graph языка Паскаль. Он состоит из 2-х пунктов меню (Выполнить, Выход). При нажатии на клавиш вверх и вниз (коды клавиш #72 и #80 соответственно) происходит поочередный выбор одного из пунктов меню (Выполнить, Выход). При нажатии клавиши Esc (код клавиши #27) происходит выход из программы. При нажатии клавиши Enter (код клавиши #13), если активно меню

«Выход», то должен произойти выход из программы, если активно меню «Выполнить», цвет этого меню должен поменяться от 1 до 15 с небольшой задержкой.

Внимание в приведенном ниже тексте реализован механизм выхода, нажатие клавиш Enter (#13) и Вниз (#80). Разобрав принцип работы программы необходимо реализовать нажатие клавиши вверх (#72).

```
Program GrIntefice;
uses crt, Graph;
var
    GD, GM, TekDeystvie, i: integer;
    Ch: char;
begin
    GD:=Detect;
    initgraph(GD,GM, '');
    TekDeystvie:=1;
    setcolor(7);
    outtextxy(240,240,'Выполнить');
    setcolor(15);
    outtextxy(240,280,'Выход');
    repeat
        Ch:=readkey;
        case ch of
        #13: begin
            case TekDeystvie of
                1: begin
                    {Выполняем какое-то действие}
                    for i:=1 to 15 do
                    begin
                        delay(3000);
                        setcolor(i);
                        outtextxy(240,240,'Выполнить')
                    end;
                    setcolor(7);
```

```
                    outtextxy(240,240,'Выполнить');
                end;
                2: Ch:=#27; {Выход}
            end;
            end;
        #80: begin
            case TekDeystvie of
                1: begin
                    TekDeystvie:= TekDeystvie+1;
                    setcolor(15);
                    outtextxy(240,240,'Выполнить');
                    setcolor(7);
                    outtextxy(240,280,'Выход');
                end;
                2: begin
                    TekDeystvie:= 1;
                    setcolor(7);
                    outtextxy(240,240,'Выполнить');
                    setcolor(15);
                    outtextxy(240,280,'Выход');
                  end;
            end;
            end;
        end;
    until Ch=#27;
    closegraph;
end.
```

2.2. Лабораторные задания

Задание 1 – Составьте блок-схему алгоритма и напишите программу на языке Паскаль, которая, построит график функции y=f(x), а также выведет в таблицу точки, по которым строился этот график. Функция берется из таблицы 1, по варианту выданному

преподавателем.

Таблица 1

Вариант	Функция	Интервал изменения X	Шаг
1	y=sin(x)	[-1,1]	0,1
2	y=cos(x)	[-1,1]	0,1
3	$y=x^2$	[-10,10]	1
4	$y=x^3$	[-4,4]	1
5	$y=x^2+1$	[-10,10]	1
6	$y=x^3+1$	[-4,4]	1
7	$y=\sin^2(x)$	[-1,1]	0,1
8	$y=\cos^2(x)$	[-1,1]	0,1
9	y=sin(x)+cos(x)	[-1,1]	0,1
10	$y=\sqrt{x}$	[0,100]	10

Задание 2 – Составьте блок-схему алгоритма и напишите программу на языке Паскаль, которая, построит объемную круговую диаграмму. Отношение секторов круговой диаграммы берется из таблицы 2, по варианту выданному преподавателем.

Таблица 2

Вариант	Сектор 1, град	Сектор 2, град
1	20	340
2	40	320
3	60	300
4	80	280
5	100	260
6	120	240
7	140	220
8	160	200
9	180	180
10	90	270

Задание 3 – Составьте блок-схему алгоритма и напишите программу на языке Паскаль реализующую механизм графического интерфейса. Меню должно содержать 4 пункта – *«Расчет», «Загрузить», «Сохранить», «Выход»*. Навигация по пунктам должна осуществляться клавишами *вверх* и *вниз*, при этом текущий пункт меню должен каким-либо образом выделяться (изменения цвета по отношению к другим пунктам, выделение цветным прямоугольником и т.п.). Активация выбранного пункта меню должна осуществлять клавишей *Enter*, при этом в служебной строке должна появляться надпись, которая будет соответствовать выбранному действию. Т.е., при выборе пункта *«Загрузить»* должна появиться надпись *«Производится загрузка из файла»*, при выборе пункта *«Расчет»* должна появиться надпись *«Производится расчет»*.

2.3 Контрольные вопросы для отчета работы

1. Перечислите основные процедуры и функции для изменения цвета линий и точек в графическом режиме языка Паскаль?
2. Перечислите основные процедуры и функции для изменения стиля заполнения в графическом режиме языка Паскаль?
3. Каково назначение и синтаксис стандартных процедур ***Bar*** и ***Bar3D***?
4. Каково назначение и синтаксис стандартных процедур ***FillPoly*** и ***PieSlice***?
5. Перечислите основные процедуры и функции для сохранения и вывода изображений в графическом режиме языка Паскаль?
6. Какая стандартная процедура языка Паскаль позволяет быстро и удобно рисовать графики функции в графическом режиме?
7. Какая стандартная процедура языка Паскаль позволяет быстро и удобно рисовать круговые диаграммы в графическом режиме?
8. Какие основные процедуры и функции языка Паскаль были использованы Вами при создании графического интерфейса?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Фаронов Валерий Васильевич. Турбо Паскаль 7.0: Начальный курс: Учебное пособие / В.В. Фаронов. – 7-е изд., перераб. – М.: Нолидж, 2002. – 576 с.: ил.

2. Фаронов Валерий Васильевич. Турбо Паскаль: Учебное пособие. – СПб.: Питер, 2007. – 367 с.: ил.

3. Информатика: Базовый курс: Учеб. пособие для втузов / Под ред. С.В. Симоновича. СПб.: Питер, 2003. – 640 с.: ил.

4. Архангельский А.Я. Программирование в Delphi: Учебник по классическим версиям Delphi. – М.: Бином, 2006. – 1152 с.: ил.

СОДЕРЖАНИЕ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
к выполнению лабораторных работ № 3-4 по дисциплине «Информационные технологии» для студентов направления 11.03.03 «Конструирование и технология электронных средств» (профиль «Проектирование и технология радиоэлектронных средств»)
всех форм обучения

Составители:
д-р. техн. наук М.А. Ромащенко,
канд. техн. наук А.А. Пирогов,
И.В. Свиридова

Компьютерный набор М.А. Ромащенко

Подписано к изданию _____
Уч.-изд. л. _____

ФГБОУ ВО «Воронежский государственный технический университет»
394026 Воронеж, Московский проспект, 14