МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Информатика

Методические указания
по выполнению лабораторных работ
Часть 2

ИЗДАТЕЛЬСТВО
Иркутского государственного технического университета
2005

Информатика. Методические указания по выполнению лабораторных работ.Ч.2.Составители: З.А.Бахвалова, И.А.Серышева.- Иркутск. Изд-во ИрГТУ,2005.-32с.

Приведены требования для выполнения лабораторных работ по курсу «Информатика». Методические указания включают три лабораторные работы по темам: массивы, записи и работа с графикой. Каждая работа содержит необходимые основные понятия и пример её выполнения. Предназначено студентам специальностей: 220200 – «Автоматизированные системы обработки информации и управления», 071900 –« Информационные системы в наукоёмких технологиях».

Лабораторная работа №1. Работа с массивами

Цельработы :

1. Знакомство со средой программирования Delphi.
2. Знакомство с формой и компонентами.
3. Организация ввода - вывода данных с использованием компонентов.
4. Создание проекта, компиляция и создание приложения, работающего в операционной системе Windows.

Требования к выполнению работы.

1. При выполнении лабораторной работы студент должен научиться вводить (выводить) данные, используя компоненты среды программирования Delphi.
2. Данные должны вводится (выводится) в поля ввода(вывода), т.е. с использованием компонентов предназначенных для ввода - вывода значений, и в таблицу.
3. Управление приложением осуществляется кнопками управления.

Основные понятия

Компонент - это готовые элементы, библиотека которых поставляется вместе со средой программирования. В качестве строительной площадки служит форма – окно Windows.

Форма - окно, которое появляется при запуске приложения, т.е это та основа, на которой размещаются другие компоненты Delphi.

Проект - состоит из форм, модулей, установок параметров проекта, ресурсов и т.д.

Файл проекта - содержит код на языке Object Pascal и является главной частью вашего приложения, с которого начинается выполнение вашей программы. Имя, которое вы даете файлу проекта, становится также именем исполняемого файла. Этот файл создается и модифицируется Delphi автоматически в процессе разработки вашего приложения и содержит лишь несколько строк.

Модуль - это исходный файл программного кода, имеющий расширение ***.pas**.

Файл модуля содержит описание стартовой формы приложения, поддерживающих ее работу процедур и модули используемые для построения компонент. В Delphi каждой форме соответствует свой модуль.

Событие - действие, возникающее в момент выполнения программы.

Обработчик события этот фрагмент программы (последовательность операторов), в которых программист указывает, что должна делать программа в ответ на действия пользователя.

Свойства компонентов, используемые в данной программе:

Caption	Название компонента
Height	Высота компонента
Width	Ширина (длина) компонента
Left	Расстояние от левой границы формы до компонента
Top	Расстояние от верхней границы формы до компонента
AutoSize	Автоматическое изменение размера компонента
WordWrap	Размещение информации в компоненте в несколько строк.
DefaultColWidth	Ширина ячейки таблицы по умолчанию

DefaultRowHeight	Высота строки таблицы по умолчанию
FixedCols	Количество фиксированных (серых) колонок в таблице
FixedRows	Количество фиксированных (серых) строк в таблице
ColCount	Количество колонок в таблице
RowCount	Количество строк в таблице
Options.goEditing	Открываем таблицу для редактированияю

Порядок выполнения работы.

Задание: Создать массив и вывести его.

Создание приложения

1. 1.Создать форму.

В верхний левый угол формы положить компонент **Label1**.
Справа от него положить компонент **Edit1**.
Ниже положить компонент **StringGrid1**, находящийся на странице компонентов **Additional.**
Справа от этих компонентов положить два компонента **Label2** и **Label3**
Внизу формы положить четыре кнопки **Button1, Button2, Button3, Button4**.
У каждого компонента установить следующие свойства:

Form

Caption	Работа с матрицей
Height	Размер установите
Width	сами

Edit1

Text	Удалить информацию из этого поля
Height	20
Width	50
Left	100
Top	5

Label1

Caption	Размер матрицы
Left	10
Top	10

Label2

Caption	Результат
Left	280
Top	10

Label3

Caption	Удалить информацию из этого поля
Left	280
Top	40
AutoSize	False
WordWrap	True

StringGrid1

FixedCols	0
FixedRows	0
ColCount	2
RowCount	2
Left	10
Top	40
Height	130
Width	250
DefaultColWidth	40
DefaultRowHeigh	20
Options.goEditing	True

Button1	
Caption	Размер
Left	70
Top	185
Height	25
Width	75

Button2	
Caption	Расчет
Left	159
Top	185
Height	25
Width	75

Button3	
Caption	Вывод
Left	247
Top	185
Height	25
Width	75

Button4	
Caption	Выход
Left	336
Top	185
Height	25
Width	75

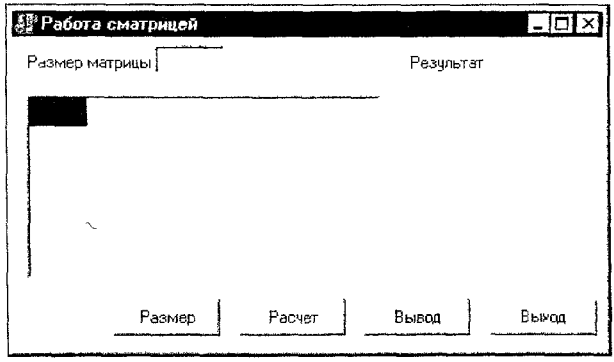


Рис 1. Вид формы в режиме редактирования

Создание кода программы

2.1 Создадим событие Button1Click (Sender:Tobject); для этого дважды щелкнем по кнопке Button1 или откроем **Инспектор Объектов** закладка **Evens** и щелкнем мышью справа от события OnClick. Delphi автоматически создаст процедуру

```
procedure TForm1.Button1Click(Sender: TObject);
begin
end;
```

Используя это событие (процедуру) изменим количество строк и столбцов **таблицы** StringGrid1 в зависимости от введенных значений в поле **Edit1.Text.**

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  StringGrid1.ColCount:=StrToInt(Edit1.Text);
  StringGrid1.RowCount:=StrToInt(Edit1.Text);
end;
```

2.2. Создадим событие Button2Click (Sender:Tobject); Delphi автоматически создаст процедуру

```
procedure TForm1.Button2Click(Sender: TObject);
begin
end;
```

Используя это событие, сформируем в памяти исходный массив. Формировать массив будем следующим образом:
В разделе описания переменных модуля **Unit1** описать массив

```
var
    Form1: TForm1;
    Mas:array[1..20,1..20] of integer;
```

- В **Procedure TForm1.Button2Click(Sender: Tobject)**, формирующей исходный массив описать переменные цикла, необходимые для формирования двумерного массива.

```
procedure TForm1.Button2Click(Sender: TObject);
var  i,j:byte;
begin
end;
```

- Сформируем массив **mas**.

```
procedure TForm1.Button2Click(Sender: TObject);
var i,j:byte;
begin
  for i:=1 to StrToInt(Edit1.Text)do
   for j:=1 to StrToInt(edit1.Text) do
   mas[i,j]:= StrToInt(StringGrid1.Cells[i-1,j-1]);
end;
```

2.3. Создадим событие вывода матрицы в компонент **Label3.**

```
procedure TForm1.Button3Click(Sender: TObject);
var i,j:byte;
begin
   for i:=1 to StrToInt(Edit1.Text) do
       for j:=1 to StrToInt(Edit1.Text) do
         Label3.Caption:= Label3.Caption +
         IntToStr(mas[i,j])+' ';
// Формируем строку сообщения, которая будет выведена в свойстве Caption
//компонента  Label3, устанавливаем ' ' чтобы элементы массива
//не сливались.
       Label3.Caption:= Label3.Caption+chr(13);
// текст будет выведен в несколько строк, для этого используется функция
// Chr, которая возвращает символ, код которого равен 13.
end;
```

2.4 Создадим событие выхода из **приложения.**

```
procedure TForm1.Button4Click(Sender: TObject);
begin
  Application.terminate;
end;
```

Варианты заданий для самостоятельной работы

Дана квадратная матрица порядка n. Найти наибольшее из значений элементов, расположенных в заштрихованной части матрицы:

1. 

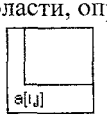
2. 

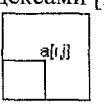
3. 

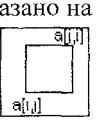
4 

5. 

Дана квадратная матрица A порядка n. Получить квадратную матрицу B порядка n, элемент b[i,j] которой равен сумме элементов матрицы A, расположенных в области, определяемой индексами [i,j] так, как показано на рисунках:

6. 

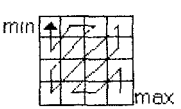
7. 

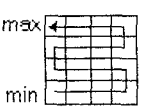
8. 

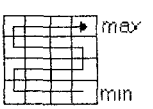
9 Перемножить две матрицы любой размерности, предварительно проверяя на согласованность.

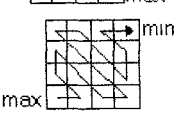
10 Дана матрица A (m,n). Найти обратную матрицу.

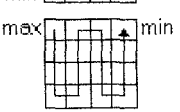
Дана квадратная матрица порядка n. Получить новую матрицу, сортируя и переставляя элементы исходной матрицы по следующей схеме:

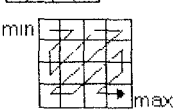
11 

12. 

13. 

14 

15. 

16. 

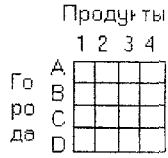
17 Вычислить методом Гаусса определитель n-го порядка, выводя при этом преобразованную к треугольному виду матрицу.

18 Сформировать из квадратной матрицы A порядка 2*n матрицу B помещая отрицательные элементы исходной матрицы A в нечетные столбцы, положительные – в четные.

19 Переписать элементы матрицы A в матрицу B, исключая из нее все повторяющиеся элементы, заменяя их 0.

20 Из матрицы A сформировать матрицу B, отсортировав по убыванию все элементы матрицы A и располагая их в матрице B по спирали, начиная с максимального элемента b_{11}.

21 В населенные пункты A, B, C,… необходимо доставить товары 1-го, 2-го и т.д. видов в количестве, задаваемом матрицей А-поставки. Стоимость перевозки единицы продукта каждого вида в заданный пункт определяется матрицей транспортных издержек С. Вычислить суммарные затраты для перевозки каждого вида продукта в необходимом количестве в заданный пункт и суммарные затраты по перевозке всех продуктов в каждый пункт.



22 Из матрицы А сформировать матрицу В, отсортировав по возрастанию все элементы матрицы А и располагая их в матрице В по спирали, начиная с минимального элемента B_{11}.

23 Переписать элементы матрицы А в матрицу В, исключая из нее все элементы, меньшие модуля суммы всех элементов матрицы А, заменяя их 0.

24 Переписать элементы матрицы А в матрицу В, сортируя элементы каждого столбца и располагая их в столбцах от минимального к максимальному.

25 Сформировать матрицу В из матрицы А, вычисляя элементы по следующей формуле:

$$b_{ij} = \begin{cases} a_{ij} + a_{i(j+1)}, & i \neq n,\ j \neq n \\ a_{ij} + a_{(i+1)(n-j+1)}, & i \neq n,\ j = n \\ a_{ij}, & i = n,\ j = n \end{cases}$$

26 Переписать элементы матрицы А в матрицу В, сортируя элементы каждой строки и располагая их от максимального элемента к минимальному.

27 Переписать элементы матрицы А в матрицу В, сортируя элементы каждой строки и располагая их в нечетных строках от минимального к максимальному, а в четных наоборот, от максимального к минимальному.

28 Переписать элементы матрицы А в матрицу В, вычитая из каждого элемента строки сумму всех четных чисел этой строки.

29 Переписать элементы матрицы А в матрицу В, вычитая из каждого элемента столбца сумму всех нечетных чисел этого столбца.

30 Дана матрица А порядка n. Вычислить сумму всех элементов матрицы, кратных 5.

31 Несколько цехов A, B, C,… производят товары 1-го, 2-го и.т.д. видов в количестве задаваемом матрицей А-план выпуска. Для производства деталей определенного вида требуются детали I-го, II–го, III-го, IV-го и.т.д. типа, необходимое количество которых задается матрицей В-норма деталей. Определить сколько де-

талей каждого типа требуется в каждый цех, а результаты вычислений занести в матрицу С-количество деталей по цехам.

32Проверить является ли квадратная матрица А порядка n симметрической или кососимметрической матрицей. Предусмотреть вывод транспонированной матрицы.

Лабораторная работа №2. Записи

Цель работы:

Получить навыки использования в проекте нескольких форм.
Ознакомиться с некоторыми приемами усовершенствования приложений.

Требования к выполнению работы.

Программа должна:

1. создать форму для ввода исходных данных,
2. использование нескольких форм
3. предусмотреть вывод исходных данных,
4. проводить проверку вводимых данных,
5. предусмотреть редактирование данных (добавление, удаление и изменение данных),
6. использовать только одну, универсальную, процедуру обработки события для нескольких компонентов
7. управление приложением осуществлять командами строки меню.
8. использовать меню (главного и контекстного)
9. производить требуемые в условии расчеты,
10. результаты расчетов и поиска должны быть оформлены в соответствующем виде (форме)

Обязательно наличие в программе:

- раздела «Справка», в этом разделе требуется описать порядок работы с программой,
- раздела «О программе», в этом разделе указать версию программного продукта, дату внесения последних изменений в программе и координаты автора.

Основные понятия.

Приемы создания меню.

Создание основной строки меню.

1.1. ***Клавиши быстрого выбора***. Перед такой клавишей в свойстве Caption поставить символ '&' – *амперсанд,* буква, перед которой стоит этот символ, в строке меню будет подчеркнута.

1.2. ***Разделительная черта,*** отделяющая группы подопций. Назовите очередную опцию символом '-' тире.

1.3. ***Разветвленные меню (дополнительные).*** Справа от подопции, открывающей дополнительное меню, в свойстве **Caption** ввести **CTRL** + →, или вызвать контекстное меню подопции и выбрать **Create Submenu**. В меню справа от этой подопции появиться ▶

1.4. *Флажок* – ***Установить (галочка)*** . Свойству **Checked**, выбранной подопции, присвоить значение ***True.***

1.5. *Флажок* – ***Выбор (кружок).*** Свойство **RadioItem** определяет зависит ли данная подопция от выбора других подопций в той же группе **GroupIndex**. Только одна подопция в группе может иметь ***True*** в свойстве **Checked.**

1.6. ***Недоступная опциия***. Свойство **Enabled** этой опции имеет значение ***True.***

Создание контекстно - зависимого меню

1. Положить компонент **PopupMenu** на форму.
2. Процесс создания и свойства контекстного меню ничем не отличаются от **TMainMenu,** поэтому создается точно также как и основное меню.
3. Чтобы связать щелчок правой кнопки мыши на компоненте с раскрытием контекстно – зависимого меню, в свойство **PopupMenu** компонента необходимо поместить имя компонента – меню.

Порядок выполнения работы

Задание:

Создать массив записей, содержащий сведения об учениках класса: фамилия, имя, отчество, домашний адрес: улица, дом, квартира, дата рождения: день, месяц, год. Написать программу, выдающую информацию по всем возможным запросам.

Создание приложения

1. Создать основную форму

Form1.

Caption | Класс

В правый верхний угол формы положить компонент **MainMenu1** (панель компонентов **Standart).** На форме Delphi разместит данный компонент в виде пиктограммы (во время работы программы данные пиктограммы невидимы, поэтому месторасположения пиктограммы роли не играет). На форму можно положить сколько угодно объектов этого класса, но отображаться в полосе меню будет только тот из них, который указан в свойстве **Меню** формы. Для того чтобы сформировать нужное вам меню или дважды щелкните мышью на пиктограмме **MainMenu1**, или в свойстве **Items** в правом столбце откройте кнопку с тремя точками. Откроется окно - конструктор меню ***Menu Designer,*** и будет выделено то место, куда должен вводится новый пункт меню. В окне **Инспектор объектов** в свойство **Caption** введите название опции меню. Каждая опция главного меню может раскрываться в список подопций или содержать конечную команду. Для создания подопции щелкните мышью по строке ниже опции и введите первую подопцию. Продолжите ввод, пока не будет введен весь список подопций, после чего щелкните по пустому прямоугольнику справа от первой опции и введите вторую опцию.

Каждая опция меню имеет свое имя **N(номер по порядку): MenuItem,** которое формируется автоматически, при создании новой опции меню. Сформировать следующее меню: опция -**Работа** (подопции - Добавить, Просмотр, Выход), опция – **Правка** (подопции –Редактировать, Удалить), опция - **Поиск** (подопции - Фамилия, Адрес, Дата рождения), опция -**Справка** (подопции - Справка, О программе).

MainMenu1

Имя	**Свой-**	**Значение**
N1	**Caption**	**&Работа**
N2	Caption	&Добавить
N3	Caption	&Просмотр
	Enabled	False
N4	Caption	-

N6	Caption	&Выход
N7	**Caption**	**&Правка**
N8	Caption	&Редактировать
	Enabled	False
N9	Caption	&Удалить
	Enabled	False
N10	**Caption**	**П&оиск**
N11	Caption	&Фамилия
	Enabled	False
N12	Caption	&Адрес
	Enabled	False
N13	Caption	&Дата рождения
	Enabled	False
N14	**Caption**	**&Справка**
N15	Caption	С&правка
N16	Caption	&О программе

Ниже меню на форму положить компонент **Label1.**

Label1.

Caption	Список класса	
Visible	False	Делает компонент **невидимым при открытии формы**

Еще ниже положить компонент **StringGrid1.**

StringGrid1.

Visible	False
Height	155
Width	355
FixedRows	1
ColCount	6

Для организации поиска на форму положить компонент **FindDialog** (страница **Dialogs**). Стандартное диалоговое окно **FindDialog** используется для поиска фрагмента текста.

FindDialog1.

FindText		Указывает образец для поиска.

Работа со стандартными диалоговыми окнами происходит в три этапа:

1 этап. На форму помещается соответствующий компонент и происходит настройка его свойств (компонент - диалог в момент конструирования программы не виден).

2 этап. Происходит вызов стандартного метода **Execute,** который создает и показывает настроенное окно диалога на экране. Вызов этого метода происходит внутри какого – либо обработчика события. Диалоговые окна являются модальными окнами, поэтому после обращения к методу **Execute** дальнейшее выполнение программы приостанавливается до закрытия этого окна. **Execute** – логическая функция, она возвращает в программу True, если результат диалога с пользователем был успешным.

3 этап. Проанализировав программа может выполнить третий этап- использование введенных с помощью диалогового окна данных (найденного текста и т.д.).

2 Создание второй формы

Нажать кнопку **NewForm** на панели инструментов ***Быстрые кнопки.***

Form2.

Caption	Ввод данных
Height	230
Width	360

Расположить на форме компоненты, как показано на рис.2

Label1 **Edit1**

Caption Фамилия

Label2 **Edit2**

Caption Имя

Label3 **Edit3**

Caption Отчество

GroupBox1. Этот компонент представляет собой контейнер для размещения дочерних компонентов. Обычно с его помощью выделяется группа управляющих элементов, объединенных по функциональному назначению.

Caption Домашний адрес

Label4 **Edit4**

Caption Улица

Label5 **Edit5** **Label6** **Edit6**

Caption Дом Caption Квартира

GroupBox2.

Caption Дата рождения

Label7 **Edit7** **Label8** **Edit8**

Caption Год Caption Месяц

Label9

Caption День

В нижний правый угол положить кнопку.

Button1.

Caption Ввод

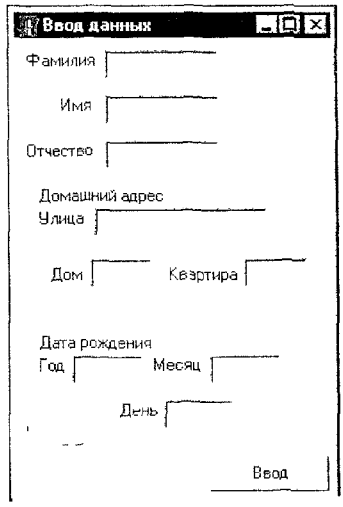


Рис 2 Форма "Ввод данных"

3. Создание третьей формы

Выберите меню **File \ New** откроется репозитарий шаблонов **NewItems**. На закладке **Forms** выберите форму **About Box**, нажмите кнопку **OK.** Шаблон окна «**About Box**» обладает всем необходимым для простого окна диалога: полем Image (изображение), куда вы можете поместить пиктограмму вашей программы; поля **Label** (меток) для имени продукта, номера версии, сообщения копирайт и комментариев, и кнопки **Ok**.

Шаблон также устана вливает некоторые из свойств окна **About.** Свойство **BorderStyle** уже

установлено в **bsDialog**, а свойства **Width** и **Height** так же имеют определенные значения. Для создания своего окна вы должны добавить в соответствующие места Label (метки) и заполнить их специфичной для вашей программы информацией. Изменить название окна.

About Box (Form3.)

Caption	О программе

4. Создание четвертой формы

Нажать кнопку **NewForm** на панели инструментов ***Быстрые кнопки***.

Form3.

Caption	Справка	
BorderIcon	biSystemMenu. =True (имеется кнопка вызова системного меню); biMinimize :=False (кнопка минимизации) biMaximize :=False (кнопка максимизации) biHelp :=False (кнопка помощи)	Определяет наличие кнопок в заголовке окна.
BorderStyle	BsDialog –рамка диалогового окна, окно не может изменять свои размеры.	Определяет стиль рамки окна

На форму положите компонент **Memo1.**

Memo1.

Align	AlClient – размещает компонент по всей области контейнера.	Указывает способ выравнивания размещаемого компонента относительно того контейнера, в котором он размещается. Пустая форма является контейнером, в котором размещаются другие компоненты.
Lines	Убрать текст в поле компонента	Содержит редактируемый текст

5. Создайте текстовый файл, который будет являться инструкцией для работы с вашей программой. Для этого откройте меню **File \ New** откроется репозитарий шаблонов **NewItems**. На закладке **New** выберите объект **Text**. Этот объект предназначен для ввода и редактирования текста. В окне редактора кода появиться новая закладка с именем **File1.txt** и с чистым полем, в которое вы можете вводить свой текст. После ввода текста сохраните свой документ. Для того, чтобы этот текст являлся частью проекта, вы можете подключить этот файл к проекту меню **Project \ Add to Project** откроется окно **Add to Project** если ваш файл не виден в окне измените поле **Тип файла** на "показать все файлы", после этого откройте файл с текстом.

Создание кода программы

Для решения данной задачи следует описать исходные переменные.

В модуле **Unit1** в разделе описания глобальных переменных описать **исходную** запись.

```
Type
 TDay = record                    //день рождения
   Day : byte;
 month : String[10];
```

```
  year : word;
end;
  TAdress = record                    //адрес
       Strit : string[20];
        home : Byte;
      Nomber : byte;
     end;

    TClas = record                    //школьник
      Fam : String[20];
     Name : String[15];
  SecName : String[20];
      Day : TDay;
   Adress : TAdress;
   end;
Var
    Form1: Tform1;
     Clas: TClas;
   ArrClas : array[1..40] of TClas;
      Col : byte;                     //количество записей
```

Обработчики событий

Form1.

При создании формы **Form1** мы на нее положили компонент **StringGrid1**. Ячейки первой, зафиксированной строки таблицы, используются в качестве заголовков колонок таблицы. Во время создания формы приложения описать эти заголовки нельзя, так элементы таблицы доступны только во время работы программы. Поэтому значения соответствующие первой строке таблицы, устанавливает процедура обработки события **OnCreate.** Это программно-управляемое событие, оно возникает независимо от действий пользователя, в определенный момент выполнения программы. Событие **OnCreate** возникает при выполнении программы в момент формирования формы в оперативной памяти.

1. Создать событие OnCreate. Открыть закладку **Events** инспектора объектов, дважды щелкнуть мышью справа от события **OnCreate**. В коде программы будет создана процедура

TForm1.FormCreate(Sender: TObject); ввести в нее следующий код:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
//Происходит формирование заголовка таблицы
StringGrid1.Cells[0,0]:='Фамилия';
StringGrid1.Cells[1,0]:='Имя';
StringGrid1.Cells[2,0]:='Отчество';
StringGrid1.Cells[3,0]:='Дата рождения';
StringGrid1.Cells[4,0]:='Адрес';
//массив записей не имеет значений и счетчик массива выставлен в 1.
Col:=0;
end;
```

2. Открыть меню **Работа** и щелкнуть мышью по опции **Добавить и ввести** следующий код.

```
procedure Tform1.N2Click(Sender: TObject);
begin
Form2.Caption:='Ввод данных'
Form2.Show;   //Метод Show показывает форму Form2 на экране
Form1.Hide;   //Метод Hide убирает с экрана форму Form1
end;
```

В нашем случае из модуля **Unit1**, соответствующего форме **Form1**, открывается другое окно (**Form2**), чтобы выполнить эту операцию модуль **Unit1** должен знать о существовании другого окна (**Form2**). Для этого на этапе конструирования (перед компиляцией) нужно активизировать окно, соответствующее модулю **Unit1**, щелкнув по нему мышью, после чего открыть меню **File \ Uses Unit.** В появившемся диалоговом окне выбрать модуль **Unit2,** соответствующий форме **Form2** и нажать **Ok.** При этом вставляется ссылка в предложение **Uses**, стоящее за зарезервированным словом **Implementation**. Точно также можно сослаться в модуле второго окна на модуль первого окна, активизируйте модуль второго окна и повторите действия. Если вы забудете сослаться на модуль, который подключен к проекту, **Delphi** при первой же трансляции программы сообщит об этом и предложит вставить недостающую ссылку.

3. Обработчик события подопции **Просмотр.**

Опция **Просмотр** в данный момент не доступна, потому что просматривать на данном этапе работы нашего приложения нечего. Как только мы в массив запишем хотя бы одну запись, опция **Просмотр** должна стать доступной (см. обработчик события `procedure TForm2.Button1Click(Sender:TObject)`).

Обработчик события опции `Просмотр`.

```
procedure TForm1.N3Click(Sender: TObject);
begin
//Станут доступными подопции меню Правка \Редактировать и
//Правка \ Удалить
Form1.N8.Enabled:=true;
Form1.N9.Enabled:=true;
StringGrid1.Visible:=True;      //Таблица станет видимая
//считать записи из массива и показать их в соответствующих ячейках
//таблицы
.....
end;
```

4. Обработчик события подопции **Выход.**

```
procedure TForm1.N4Click(Sender: TObject);
begin
Application.Terminate;
//Закрывает приложение и выгружает его из памяти.
end;
```

4. Обработчик события подопции **Правка\ Редактирование.**

Опция **Правка \ Редактирование** в данный момент не доступна, потому что редактировать на данном этапе работы нашего приложения нечего. Как только мы в массив запишем хотя бы одну запись, опция **Правка \ Редактирование** должна стать доступной (см. обработчик события **procedure TForm1.N3Click(Sender: TObject)**).

```
procedure TForm1.N7Click(Sender: TObject);
begin
// Номер выделенной строки в таблице содержит
// свойство StringGrid1.Row : LongInt;
//Считать этот номер и сформировать запись из значения полей этой строки
.....
//в окне Form2 вывести значения полей записи в поля Edit.
Form2.Caption := 'Редактирование';
Form2.Show;
With Form2 do
Begin
Edit1.Text:=...
End;
 end;
```

5. Обработчик события подопции **Правка \ Удалить.**

Опция **Правка \ Удалить** в данный момент не доступна, потому что удалять на данном этапе работы нашего приложения нечего. Как только мы в массив запишем хотя бы одну запись, опция **Правка \ Удалить** должна стать доступной (см. обработчик события **procedure TForm2.Button1Click(Sender:TObject)**).

В этом обработчике событий удаляется выделенная строка. Номер выделенной строки в таблице содержит свойство **StringGrid1.Row : LongInt;**

```
procedure TForm1.N8Click(Sender: TObject);
begin
//начиная со строки, которую надо удалить в таблице и в массиве,
//передвинуть строки на одну вперед.
end;
```

6. Обработчики события опции **Поиск**

Подопции **Поиск** в данный момент не доступны, потому что проводить поиск на данном этапе работы нашего приложения нечего. Как только мы в массив запишем хотя бы одну запись, подопции **Поиск** должны стать доступными (см. обработчик события **procedure TForm2.FormCloseQuery(Sender: TObject; var CanClose: Boolean)**.

Работа со стандартным окном **FindDialog** проходит в три этапа. Первый этап происходит на этапе разработки программы.

На втором этапе происходит вызов стандартного метода **Execute,** который создает и показывает настроенное окно **FindDialog** на экране. Вызов этого метода происходит внутри обработчиков событий каждой подопции **Поиск.**

Для определения, какой подопцией было открыто окно поиска используем глобальную переменную **Clas** описанную в модуле **Unit1** в разделе **Var.**

Создайте обработчик события подопции **Поиск \ Фамилия** (в режиме конструктора меню щелкнуть мышью по нужному пункту)

```
procedure TForm1.N10Click(Sender: TObject);
begin
if FindDialog1.Execute then
     Clas.Fam:= FindDialog1.FindText;
end;
```

на третьем этапе компонент **FindDialog** реализует возможность поиска введенного в него фрагмента текста. С этой целью для компонента определено событие **OnFind**, которое возникает всякий раз, когда пользователь нажимает кнопку **Найти**. Создайте событие **TForm1.FindDialog1Find(Sender: TObject);** для этого выделите компонент **FindDialog1**, откройте закладку **Events** Инспектора объектов и дважды щелкните в поле справа от события **OnFind**.

```
procedure TForm1.FindDialog1Find(Sender: TObject);
begin
//Прочитать массив и сравнить поле Fam каждой записи массива
//с введенной переменной,
//Если такая запись существует, показать ее
//в противном случае выдать сообщение, что такой записи нет.
end;
```

Таким же образом создается обработчик события для подопций **Поиск \ Адрес** и **Поиск \ День рождения.**

7. Обработчик события подопции **Справка \ Справка.**

```
procedure TForm1.N14Click(Sender: TObject);
begin
Form3.ShowModal;
// Выводит на экран Form3 (Справка) и делает ее модальной.
end;
```

8. Обработчик события подопции **Справка \ О программе.**

```
procedure TForm1.N15Click(Sender: TObject);
begin
//Форма имеет имя AboutBox1, а метод ShowModal показывает форму
// и не дает возможности переключиться на другую форму,
//пока не закроете эту формулу.
AboutBox1.ShowModal;
end;
```

Form2

1. **Обработчик события** кнопки **Button1** формы **Form2**.

```
procedure TForm2.Button1Click(Sender: TObject);
begin
If Caption='Ввод данных'
        then Col:=Col+1;
```

```
//Сформировать массив
ArrClas[Col].Fam:=Edit1.text;
ArrClas[Col].Name:=Edit2.text;
ArrClas[Col].SecName:=Edit3.text;
ArrClas[Col].Adress.Strit:=Edit4.text;
ArrClas[Col].Adress.home:=StrToInt(Edit5.text);
ArrClas[Col].Adress.Nomber:=StrToInt(Edit6.text);
ArrClas[Col].Day.Day:=StrtoInt(Edit9.text);
ArrClas[Col].Day.month:= Edit8.text;
ArrClas[Col].Day.year:=StrToInt(Edit7.text);
//Проверить во все ли поля введены данные,
//если в поле данные не введены выдать сообщение.
// Очистить поля Edit
Edit1.text:='';
Edit2.text:='';
Edit3.text:='';
Edit4.text:='';
Edit5.text:='';
Edit6.text:='';
Edit7.text:='';
Edit8.text:='';
Edit9.text:='';
end;
```

2. **Обработчик события закрытия формы**

Обработчик события **CloseQuery** вызывается, когда предпринимается попытка закрыть форму. Это происходит при вызове метода **Close** или когда пользователь нажимает кнопку **Закрыть** в заголовке формы. В этой процедуре параметр **CanClose** определяет разрешено ли закрытие формы. Если **CanClose = true** форма закрывается.

```
procedure TForm2.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
//проверить поля Edit пустые они или нет,
//если поля не пустые выдать сообщени о том, что запись не сохранена.
.....
If MessageDlg('Вы действительно хотите закрыть это окно?',mtCustom,[mbOk,mbCancel],0)=mrOk
  then
    begin
    CanClose :=true;
//Сделать видимыми все пункты меню
    Form1.N3.Enabled:=true;
    Form1.N10.Enabled:=true;
    Form1.N11.Enabled:=true;
    Form1.N12.Enabled:=true;
    end
```

```
   else CanClose:=False;
end;
```

3. Обработчик события нажатия клавиши клавиатуры

Обработчик события **OnKeyPress** – вызывается, когда пользователь нажимает клавишу с «читаемым» символом из набора ASCII. Клавиши не имеющие символьного значения (например: **SHIFT** или **F1**) не генерируют событие **OnKeyPress**. Для того, чтобы создать это событие выделите компонент, для которого создается это событие и используя закладку Events Инспектора Объектов создать обработчик события.

```
procedure TForm2.Edit1KeyPress(Sender: TObject; var Key:
Char);
begin
//проверяется какая клавиша на клавиатуре нажата
case key of
'A'..'Я','a'..'я':;
                #13:begin
                    if Sender=Edit1 then Edit2.Setfocus;
                    if Sender=Edit2 then Edit3.Setfocus;
                    if Sender=Edit3 then Edit4.Setfocus;
                    if Sender=Edit4 then Edit5.Setfocus;
                    end;
            #32,#8 :;
else
  begin
    key:=chr(0);
    ShowMessage('Неправильный ввод');
  end;
end;
 end;
```

Обработчики событий **OnKeyPress** редакторов **Edit1, Edit2, Edit3, Edit4, Edit9,**отличаются только реакцией на нажатие клавиши **Enter**. Поэтому можно использовать только одну, универсальную, процедуру обработки события **OnKeyPress,** которая в случае нажатия клавиши **Enter** обеспечивала бы перевод фокуса (в данном случае – текстового курсора) в следующее поле ввода, в зависимости от того ввод в какое поле редактирования происходит.

Чтобы все редакторы для обработки этого события использовали одну процедуру надо выделить соответствующий редактор открыть закладку **Events** Инспектора Объектов и ввести имя этой процедуры в поле имени обработчика события **OnKeyPress.** Повторить эти действия для всех т полей ввода.

Параметр ***Sender*** присутствует во всех процедурах обработки событиями Delphi. Когда происходит событие, связанное с некоторым обработчиком событий, этот обработчик события получает сообщение для объекта, который вызвал это событие. Т.е. ***Sender*** содержит имя объекта, вызвавшего это событие. Так как в нашем случае обработчик события используется совместно пятью объектами, полезно знать какой объект вызывает данное событие.

Создайте обработчик события для тех редакторов, в поля которых происходит ввод значений целого типа, при этом значения в этих полях должны входить в определенные границы.

Form3.

1. Обработчик события для кнопки **Ok** формы **"О программе"**

```
procedure TAboutBox1.OKButtonClick(Sender: TObject);
begin
AboutBox1.Close; // Метод закрывает форму.
end;
```

1. Обработчик события возникающий в момент активизации (появления на экране) формы **Справка**.

```
procedure TForm3.FormActivate(Sender: TObject);
begin
//Информация из файла File.txt загружается в редактор Memo1.
Memo1.Lines.LoadFromFile('File1.txt');
end;
```

Варианты заданий для самостоятельной работы

1. Создать запись, содержащую сведения об учениках класса: фамилия, имя, отчество, домашний адрес: улица, дом, квартира, дата рождения: день, месяц, год. Написать программу, которая считает сколько человек живет на одной улице, выдающую информацию об учениках, чьи фамилии начинаются с одной буквы, найти всех учеников родившихся в один месяц и имеющих одинаковое отчество.
2. Создать запись, содержащую сведения о спортсменах: фамилия, имя, год рождения, вид спорта, во сколько лет начал заниматься спортом, звание, в каком году присвоено это звание. Написать программу, которая считает сколько спортсменов занимается спортом более определенного количества лет, вывести на экран спортсменов, занимающихся определенным видом спорта, найти всех спортсменов, имеющих одинаковое звание и занимающихся определенным видом спорта.
3. Создать запись, содержащую сведения об учениках класса: фамилия, имя, отчество, аттестация по математике, русскому языку, физике за несколько месяцев. Сформировать бланк аттестации для каждого ученика, посчитать количество не аттестованных учеников, вывести на экран список учеников, имеющих средний бал выше среднего по любому предмету и четверку по математике.
4. Создать запись, содержащую сведения о рабочих бригады: фамилия, имя отчество, год рождения, должность, разряд, пол, год поступления на работу. Написать программу, которая считает месячную зарплату рабочих бригады (минимальная заработная плата*разряд +20% северных+25% за выслугу лет, если рабочий проработал больше 3 лет –13% подоходный налог), выдать ведомость начисления зарплаты отсортированную по фамилиям, найти рабочих, чей заработок больше среднего по бригаде, определить рабочих, которые в этом году уйдут на пенсию.
5. Создать запись, содержащую сведения о магазинах, продающих кондитерские товары: название магазина, вид товара, название товара производитель товара,

цена единицы товара, вес товара, срок реализации. Написать программу, которая выводит на экран товары определенного производителя, считает, сколько магазинов продает конфеты определенного названия и определенного производителя, выдать магазины торгующие просроченным товаром.

6. Создать запись, содержащую сведения об учениках музыкальной школы: фамилия, имя, в классе какого инструмента учится, в каком году поступил в школу, в каких конкурсах участвовал. Распечатать список учеников, которые учатся играть на определенном инструменте, указать сколько лет они занимаются музыкой, найти учеников участвующих в определенных конкурсах и родившихся в одном году.
7. Создать запись, содержащую сведения о рабочих предприятия: фамилия, имя отчество, год рождения, должность, месячная заработная плата за пол года, пол, год поступления на работу. Написать программу, которая определяет рабочих, чья заработная плата меньше средней по предприятию, распечатать список женщин проработавших на предприятии больше 10 лет с указанием зарплаты, стажа работы и должности, найти рабочих занимающих определенную должность и проработавших одинаковое количество лет на этом предприятии.
8. Создать запись, содержащую сведения об учениках школы: фамилия, имя, класс, оценки за последнюю четверть. Посчитать и распечатать фамилии тех учеников, которые не получили ни одной тройки за четверть; определить класс с лучшей успеваемостью; найти учеников, определенного класса, чей средний бал за четверть меньше среднего.
9. Создать запись, содержащую сведения о детях, посещающих детский сад: фамилия, имя, группа, дата рождения: год, месяц, день. Посчитать количество детей посещающих садик; распечатать отсортированный по фамилии список детей определенной группы; найти детей одной группы, родившихся в один месяц.
10. Создать запись, содержащую сведения об учителях школы: фамилия, имя, отчество, год рождения, названия высшего учебного заведения, которое окончил учитель, год поступления на работу, предметы, которые преподает учитель, недельная нагрузка. Распечатать список учителей, которые могут преподавать определенный предмет; посчитать учителей закончивших одно высшее заведение; найти учителей, которые работают больше 10 лет и имеют среднюю недельную нагрузку больше средней по школе.
11. Создать запись, содержащую сведения об участниках олимпиады по «Информатике»: фамилия, имя, отчество, год рождения, названия высшего учебного заведения, в котором учится участник, количество баллов. Распечатать анкетные данные всех участников олимпиады, которые набрали более 30 баллов; рассчитать количество участников набравших баллы больше среднего и выдать их список; найти участников, обучающихся в одном учебном заведении и родившихся в один год.
12. Создать запись, содержащую сведения об учениках школы: фамилия, имя, отчество, класс, оценки за год, дата рождения: год, месяц, день. Распечатать отсортированный список учеников по классам, которые являются хорошистами и отличниками по итогам года; рассчитать количество учеников, чей средний

балл ниже среднего по итогам года; найти учеников имеющих тройки за год и родившихся в один месяц.

13. Создать запись, содержащую сведения об учениках класса: фамилия, имя, отчество, класс, вес, рост, пол, дата рождения: год, месяц, день. Распечатать в алфавитном порядке список учеников, чей вес ниже определенного веса; определить среднюю массу мальчиков и средний рост девочек; найти учеников, имеющих определенный вес и родившихся в определенный месяц.
14. Создать запись, содержащую сведения о переписи населения: фамилия, имя, отчество, образование, дата рождения: год, месяц, день, адрес: город, улица, дом, квартира. Распечатать анкетные данные жителей родившихся после 1990года, отсортированные по фамилии; рассчитать количество жителей проживающих в определенном городе; найти жителей имеющих определенное образование и родившихся в определенном году.
15. Создать запись, содержащую сведения об абитуриентах поступающих в институт: фамилия, имя, отчество, школа, дата рождения: год, месяц, день, факультет, оценки за экзамены. При поступлении в университет лица, получившие оценку « неудовлетворительно» на экзамене не допускаются к следующему экзамену. Написать программу, которая составляет список абитуриентов в алфавитном порядке к каждому экзамену по итогам предыдущего экзамена; посчитать количество абитуриентов, получивших оценку «неудовлетворительно» после каждого экзамена; найти абитуриента, получившего наивысшее количество баллов и поступающего на определенный факультет.
16. Создать запись, содержащую сведения о студентах факультета: фамилия, имя, отчество, группа, форма обучения, оценки за последнюю сессию. Написать программу назначения стипендии студентам по итогам сессии, используя следующие правила:

16.1. Студенты, обучающиеся на коммерческой форме обучения, стипендию не получают.

16.2. Студенты обучающиеся на бюджетной основе получают стипендию:

16.3. Если все оценки 5, назначается повышенная стипендия;

16.4. Если все оценки 4 и 5, назначается обычная стипендия;

16.5. Если есть оценка3, стипендия не назначается

17. Написать программу, которая, распечатывает список фамилий каждой группы в алфавитном порядке с пометкой о назначении стипендии; рассчитать количество студентов не сдавших сессию в срок; найти студентов одной группы и имеющих определенную оценку по каждому экзамену
18. Создать запись, содержащую сведения об учениках: фамилия, имя, класс, вес, рост, масса. Написать программу, которая выдает список классов отсортированных по росту учеников; рассчитать количество учеников, чей рост выше среднего; найти учеников, которые могут заниматься в баскетбольной секции, если баскетболист должен быть определенного возраста и определенного роста.
19. Создать запись, содержащую сведения о лекарствах, хранящихся на аптечном складе: наименование лекарственного препарата, производитель лекарственного препарата, количество, цена единицы лекарственного препарата, срок хранения (в месяцах). Написать программу, которая выводит на экран наи-

меньшую цену определенного лекарственного препарата; рассчитать сколько стоят все препараты определенного производителя; найти лекарственные препараты срок хранения, которых меньше определенного срока, а количество больше 10 и выдать их в алфавитном порядке.

20. В столовой предлагается N-комплексных обедов, состоящих из Q –блюд, создать запись, содержащую сведения о комплексных обедах столовой: номер комплексного обеда, блюдо, калорийность, стоимость. Написать программу, которая считает стоимость определенного комплексного обеда и рассчитывает количество калорий; сформировать самый низкокалорийный обед; найти самое высококалорийное блюдо.
21. Создать запись, содержащую сведения о продукции, хранящейся на торговом складе: наименование товара, цена товара до уценки, срок хранения товара, цена товара после уценки, общее количество товара. Написать программу, формирующую ведомость уценки товара по следующему правилу: если продукция хранится на складе дольше N месяцев, то она уценивается в два раза, а если срок хранения превысил M (M<N) месяцев, но не достиг N, то в 1,5 раза. Ведомость уценки товаров должна содержать следующие поля: наименование товара, цена товара до уценки, срок хранения товара, цена товара после уценки, общая стоимость товара до уценки, общая стоимость товара после уценки; рассчитать общую стоимость уцененного товара до уценки, найти товар, не подлежащий уценке с минимальной стоимостью.
22. Создать запись, содержащую сведения о спортсменах - многоборцев принимающих участие в соревнованиях по M видам спорта: фамилия, имя, звание, количество очков по всем видам спорта. По каждому виду спорта спортсмен набирает определенное количество очков. Написать программу, которая выдает данные о спортсменах, отсортированный по общему результату спортсмена; найти спортсменов, получивших наибольшее количество очков по определенному виду спорта; рассчитать разницу в очках для спортсменов, занявших первое и последнее место в определенном виде спорта.
23. Создать запись, содержащую сведения о бюро путешествий: наименование бюро (фирмы), наименования маршрутов, вид транспорта, продолжительность путешествия, стоимость путевки, ближайшая дата отправления группы. Написать программу, которая распечатывает в алфавитном порядке все сведения о путешествиях на определенном виде транспорта; рассчитать количество дней до отправления в путешествие по определенному маршруту продолжительностью до определенного количества дней; найти маршрут, продолжительность, которого не больше определенного количества дней и отправление в течении ближайшей недели.
24. Создать запись, содержащую сведения об автогонках: страна-организатор гонок; название гонок; фамилия гонщика; тип машины количество стран, по которым проходит гонка; страна-участница; дата начала гонок; дата окончания гонок; призовой фонд. Написать программу, которая распечатывает данные об автогонщиках в алфавитном порядке; рассчитать количество гонок, проходящих зимой; найти гонки с максимальным призовым фондом, в которых не участвует Россия.

25. Создать запись, содержащую сведения об экзаменационной ведомости: предмет, номер группы, номер зачетной книжки, фамилия, имя отчество студента, его оценки по итогам текущей сессии. Написать программу распечатывающую список студентов в алфавитном порядке; рассчитать количество студентов сдавших экзамен на определенную оценку; найти студентов с одинаковыми отчествами и одинаково сдавших сессию.
26. Создать запись расписание вычислительного зала, содержащую следующую запись: день недели, название предмета, время начала и конца пары (час, минута), фамилия преподавателя. Написать программу, выводившую полную информацию о занятиях в зале отсортированную по времени (дата, час, минута); посчитать количество занятий в часах по определенному предмету; найти фамилию преподавателя, преподающего определенный предмет, чаще всех занимающегося в этом зале
27. Создать запись, содержащую сведения о книгах библиотеки: название, вид издания, год выпуска (для книги), дату выпуска (для газет и журналов), автор (для книги), редактор (для газет и журналов), объем, количество. Написать программу, которая распечатывает информацию об изданиях, вышедших в этом году отсортированных по виду издания и названию; рассчитать количество книг в библиотеке; найти определенную газету или журнал, вышедшие в определенное время.
28. Создать запись, содержащую сведения о театрах: город, название театра, название спектакля, жанр, дата выпуска спектакля, цена билета. Написать программу, которая распечатывает данные о театрах отсортированные по жанру спектакля; найти театр имеющий максимальную цену билета на мюзикл; рассчитать количество спектаклей определенного жанра, выпущенные в определенное время.
29. Создать запись, содержащую сведения о магазине вычислительной техники: название магазина, тип процессора, страна-изготовитель, цена единицы, поступило штук, продано штук. Написать программу, которая распечатывает все данные о магазине, отсортированные по типу процессора и стране-изготовителю; рассчитать цену определенного типа компьютера определенного производства; найти магазины, торгующие вычислительной техникой определенного производителя и продавших определенное количество машин.
30. Создать запись, содержащую сведения об аудиозаписях: авторы песни, исполнитель, страна, вид носителя (диск, кассета, пластинка), тираж, год выпуска, цена единицы. Написать программу, которая распечатывает данные отсортированные по виду носителя и исполнителю; рассчитать количество песен определенного автора с определенным тиражом; найти диски, выпущенные не позднее определенного года в определенной стране

Лабораторная работа №3. Построение графиков функций

Цель работы

Студент должен научиться:

1. создавать изображения в приложениях, манипулировать точками этих изображений.
2. создавать модуль описания процедур и функций, в котором будут использоваться описанные пользователем процедуры и функции.

Требования к выполнению работы

Программа должна

1. Графики четырех функций соответствующего варианта построить различными цветами в одном окне.
2. Каждый график построить в отдельном окне в четверть экрана.
3. Графики должны быть построены с учетом масштаба (границ по оси OX).

Исходные данные:

1. **A** и **B** границы по оси **OX**
2. **N** – число точек на графике
3. **Y1** и **Y2** – границы по оси **OY**

Исходные данные задаются самостоятельно, в соответствии с предлагаемыми функциями и вводятся с клавиатуры (необходима проверка данных при вводе).

Шаг по оси **OX** вычисляется по формуле : **H=(A+B)/N**

Алгоритм построения графика функции в виде точек;

1. Ввод значений **A,B**
2. Вычисление значений **Y1,Y2**
3. Установка цвета фона и цвета рисунка
4. Установка положения окна вывода (по умолчанию вся форма)
5. Вычисление шага по оси **OX** **H=(B-A) / N**
6. Вывод точек графика в цикле, параметром которого будет координата **X**, которая изменяется от **A** до **B** с шагом **H** .

Основные понятия

Canvas (канва, холст). Канва не является компонентом, но она входит в качестве свойства в те из них, которые должны уметь нарисовать себя и отобразить какую-либо информацию. Это область рисунка на форме и во многих других графических компонентах. Свойство **Canvas** предоставляет коду Delphi возможность изменения области рисунка во время исполнения. Особенность **Канвы** в том, что она содержит методы и свойства, существенно упрощающие работу с графикой в Delphi.

Свойства класса TCanvas.

Font : TFont;	Шрифт канвы
Pen : TPen;	Перо канвы
Brush : TBrush;	Кисть канвы
Pixels[x,y:Integer] :TColor;	Устанавливает цвет точки с координатами x,y
PenPos : TPoint;	Содержит координаты текущей точки пера канвы.

ClipRect : TRect;	Определяет область отсечения канвы. То, что при рисовании попадет за пределы этого прямоугольника, не будет изображено.

Цвета точки

Константа	**Цвет**	**Константа**	**Цвет**
clAqua	Aqua	clMaroon	Maroon
clBlack	Black	clNavy	Navy blue
clBlue	Blue	clOlive	Olive green
clDkGray	Dark Gray	clPurple	Purple
clFuchsia	Fuchsia	clRed	Red
clGray	Gray	clSilver	Silver
clGreen	Green	clTeal	Teal
clLime	Lime green	clWhite	White
clLtGray	Light Gray	clYellow	Yellow

Методы класса TCanvas.

Procedure LineTo(x,y:Integer);	
ProcedureMoveTo(x,y:Integer);	

Координаты. Все визуальные компоненты имеют свойства **Top** и **Left**. Значения, запоминаемые этими свойствами, определяют местоположение компонента на форме. Иными словами, компонент размещается в позиции (**X,Y**), где **X** – относится к свойству **Left** , а **Y** - к свойству **Top**. Значения **X и Y** выражаются в пикселах. Пиксел (точка) – это наименьшая частичка поверхности рисунка, которой можно манипулировать. Прямоугольная система координат **канвы** для отображения отдельных точек и границ фигуры имеет следующие параметры :

- Начало координат (точка [0,0]) расположена в левом верхнем углу канвы.
- Координата X отсчитывается по горизонтали слева направо.
- Координата Y отсчитывается по вертикали сверху вниз.
- Размеры канвы совпадают с размерами клиентской области **компонента** владельца.

Порядок выполнения работы.

Задание

Построить график функции F:=0.5*x+2*cos(x)

Создание приложений

1. Создать основную форму

Form1.

Caption	Построение графиков

На форму положить компонент **MainMenu1** (панель компонентов **Standart).** Сформировать следующее меню: опция- **Границы**, опция –**Один график**, опция – **Четыре графика**.

MainMenu1

Имя	**Свойство**	**Значение**
N1	Caption	&Граница
N2	Caption	&Один график
N3	Caption	&Четыре графика

2. Создание второй формы

Нажать кнопку **NewForm** на панели инструментов ***Быстрые кнопки.***

Form2.

Caption	Границы
Height	130
Width	270

Расположить на форме компоненты, как показано на Рис.3.

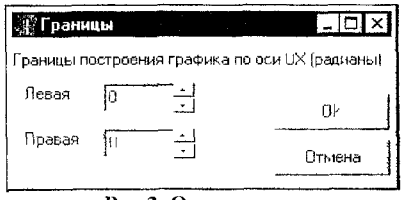


Рис 3. Окно границ

Label1

Caption - Границы построения графика по оси X (радианы).

Label2 **Edit1**

Caption - Левая

Label3 **Edit2**

Caption - Правая

На закладке **Win32** найти компонент **UpDown** и положить его справа от компонента **Edit1**.

UpDown1

Associate	Edit1
Increment	1
Max	В зависимости от функции
Min	
Position	0

Исходное показание счетчика

На закладке **Win32** найти компонент **UpDown** и положить его справа от компонента **Edit2**.

UpDown2

Associate	Edit2
Increment	1
Max	В зависимости от функции
Min	
Position	0

Button1 **Button2**

Caption - Ok Caption – Отмена

Создание кода программы

Для решения данной задачи следует описать исходные переменные.

В модуле **Unit1** в разделе описания переменных описать следующие переменные.

```
Const
     Xn=10;          //Координата  начала  оси X на форме
     Yn=10;          //Координата  начала  оси Y на форме

var
     Form1: TForm1;
          x,y : real;        //координаты пиксела
```

```
    Xmin,
    Xmax,
    Ymin,
    Ymax   : real;       //Правая и левые границы оси X
    Dx,Dy  : real;       //Шаги по оси X и Y
```

```
Function F(x:real):real;
```
//Функция описывается в разделе описания Interface, потому, что эта функци
//используется, как в модуле Unit1, где она описана, так и в модуле Unitt2

Обработчики событий

Form1

1. Описание заданной функции.

```
Function F(x:real):real;          // F(X) - Заданная функция.
begin
 F:=0.5*x+2*cos(x)
end;
```

3. Функции описывающие переход от вещественных координат к экранным

- по оси X

```
function xe(x:real):integer;
begin
 xe:=round((x-xmin)/dx)+10;
end;
```

- по оси Y

```
function ye(y:real):integer;
begin
  ye:=(Form1.Height-10)-round((y-ymin)/dy)
end;
```

3. Дважды щелкнуть мышью по меню ***Границы*** и ввести следующий код.

```
procedure TForm1.N1Click(Sender: TObject);
begin
Form2.Show;
end;
```

4. Дважды щелкнуть мышью по меню ***Один график*** и ввести следующий код

```
procedure TForm1.N2Click(Sender: TObject);
begin
```
//Рисование осей.
//Установим красный цвет пера
```
Canvas.Pen.Color:= ClRed;
```

```
//нарисуем ось OY
//Устанавливаем перо в точку X=0, Y=Ymax
Canvas.MoveTo(Xe(0),ye(ymax));        /
//Рисуем линию из точки X=0,Y=Ymax до точки X=0, //Y=Ymin
Canvas.lineTo(xe(0),ye(ymin));
//нарисуем ось OX
// Устанавливаем перо в точку X=Xmin, Y=0
Canvas.MoveTo(Xe(Xmin),ye(0));
//Рисуем линию из точки X= Xmin,Y=0 до точки X=Xmax,/Y=0
Canvas.lineTo(xe(Xmax),ye(0));

//Вывод значения левой границы оси 0X
Canvas.TextOut(xe(xmin),ye(0),Form2.Edit1.Text);
//Вывод значения правой границы оси 0X
Canvas.TextOut(xe(xmax),ye(0),Form2.Edit2.Text);
//Рисование графика
//Устанавливаем значение X равное значению левой границы
x:=xmin;
//Устанавливаем перо в точку X=Xmin, Y=F(Xmin)
Canvas.MoveTo(xe(x),Ye(F(x)));
while x<xmax do
begin
//Определяем координату Y для текущего X
y:=f(x);
//Рисуем линию из предыдущей точки до текущей
Canvas.LineTo(Xe(x),Ye(y));
//Изменяем значение X  на величину шага изменения функции по оси X
x:=x+dx;
end;
end;
```

Form2

Обработчик события кнопки **Button1** расположенной на форме **Form2**.

```
procedure TForm2.Button1Click(Sender: TObject);
begin
//Границы функции по оси X
Xmin:=StrToInt(edit1.text);      //Левая граница графика по оси X
Xmax:=StrToInt(Edit3.Text);      //Правая граница графика по оси X
//шаг изменения функции по оси X
dx:= (Xmax-Xmin)/nx;
X:=Xmin;
//Определения максимального и минимального значения исходной функции
//(границы функции по оси Y).
Ymin:=1000; Ymax:=-1000;
repeat
```

```
 y:=F(X);
 if y>Ymax then yMax:=y;
 if Y<YMin then YMin:=y;
 X:=X+dx;
until X>=Xmax;
```
//шаг изменения функции по оси Y
```
dy:=(Ymax-Ymin)/ny;
Form2.Close;
end;
```

Варианты заданий для самостоятельной работы

№ вар	F1(x)	F2(x)	F3(x)	F4(x)
1.	cos(x)	2cos(x)	cos(2x)	Cos(x/2)
2.	sin(x)	2sin(x)	sin(2x)	sin(x/2)
3.	Exp(x)	2Exp(x)	Exp(2x)	Exp(x/2)
4.	cos(x)+0.5x	2cos(x)+2x	cos(2x)-4x	3x-Cos(x/2)
5.	x^2	$-x^2$	ax^2	x^2+2
6.	3*x+sin(x)	x/3 –cos(x/3)	x+3*cos(x)	x-(cos(x))/3
7.	x^3	$2x^3$	$x^3/2$	$(x/2)^3$
8.	$\sqrt{\lvert x \rvert}$	$2x^3$	$x^3/5$	x^3+4
9.	Exp(x)	Exp(x/2)	Exp(x-1)	Exp(2x)
10.	Exp(x)+2x	x^2+2Exp(x)	Exp(2x)-0.5x	Exp(x/2)+7
11.	cos(x)	cos(x/2)	$(Cos(x))^2$	Cos(2x)
12.	sin(x)	sin(x/2)	$(\sin(x))^3$	$(Sin(x))^2$
13.	ln(x)+3	3x-2ln(x)	0.5x+ln(x/2)	3x* ln(x)/2
14.	x^3	x^3-2	$2x^3$	$2x^3-2$
15.	$-x^3$	$-x^3+2$	$-x^4/4$	$0.5*x^2-2$
16.	x^2	$-3*x^2$	$-x^2/3$	$-x^2+3$
17.	x^3	$-2x^3$	$x^3/2$	$-x^3+2$
18.	sin(x)/2	2cos(x)+3x	3*sin(2x)	3*Cos(2x)
19.	cos(x)	2sin(x)	cos(2x)	Sin(x/2)
20.	sin(x)	2cos(x)	sin(2x)	Cos(2x)
21.	lg(x)	lg(2x)	lg(x/2)	(lg(x))/2
22.	|x|	1/|X|	x^2	$(2x)^2$
23.	2sin(x)	3sin(x)	sin(3x)	Sin(2x)
24.	lg(x)	ln(x)	ln(x/2)	lg(x/2)
25.	(Cos(x))/2	Sin(x)+2	Cos(x)+3	(Sin(x))/3
26.	cos(x)	cos(x/2)	cos(x/3)	Cos(x/4)
27.	sin(x)	sin(2x)	sin(3x)	Sin(4x)
28.	cos(x)	2cos(x)	cos(x/2)	$\sqrt{\lvert \cos(x) \rvert}$
29.	2sin(x)	cos(2x)	$\sqrt{\lvert Cos(x)^2 \rvert}$	Cos(2x)

№ вар	F1(x)	F2(x)	F3(x)	F4(x)
30.	ln(x/2)	lg(x/2)	ln(x)	lg(2x)
31.	ln(x)	ln(2x)	ln(3x)	ln(4x)
32.	ln(x)	ln(2x)	$\ln(x)^2$	$(Ln(x))^2$
33.	|x|	1/|2X|	1/|3X|	1/|4X|
34.	ln(x)	2ln(x)	$\ln(x/2)$	$\ln(x)/2$
35.	x^2+4	$-x^2-4$	$x^2/4$	$4x^2$
36.	cos(x)	2cos(x)	cos(x/2)	$\sqrt{\lvert\cos(x)\rvert}$
37.	cos(2x)	Cos(x/2)	Cos(x+5)	5+cos(x)
38.	sin(2x)	sin(x/2)	Sin(x+4)	Sin(x)+4
39.	lg(2x)	lg(x)+2	lg(x-2)	(lg(x))/2
40.	ln(3x)	ln(2+x)	ln(x/2)	(ln(x))/2
41.	ln(x)	ln(2x)	ln(x/2)	ln(x)/2
42.	|3-|x-2||	|3-x| -2	3-(x-2)	3-|x|-2

Требования к выполнению лабораторных работ по курсу «Информатика»

Выполнение лабораторной работы предполагает следующие действия:

1. постановка задачи,
2. составление алгоритма,
3. выбор структуры программы,
4. подготовка тестов,
5. запись программы на алгоритмическом языке, в среде визуального программирования Delphi.
6. отладка программы,
7. тестирование программы.

В состав отчета должны входить следующие разделы:

1) Титульный лист.
2) Содержание.
3) Для каждого задания в отчете должна содержаться следующая информация:
   - постановка задачи;
   - спецификация входных и выходных глобальных параметров программы;
   - описание каждой процедуры и функции в содержательном и математическом аспектах
   - спецификация процедур и функций с указанием имен процедур и функций и списка параметров;
   - таблица сообщений;
   - блок схемы процедур
   - таблица тестов;
   - текст программы, отпечатанный на принтере и снабженный комментариями;
     - в начале программы, с указанием автора, группы, задания;

- в начале каждой процедуры или функции, с описанием того, для чего предназначена процедура и описанием параметров
- в основных (ключевых) участках программы, с указанием на семантику выполняемых действий.

- дата последних изменений (в модуле о программе).
- результаты работы программы.

Все страницы отчета должны быть пронумерованы, в его начале должно быть приведено содержание.

Используемая литература

1. Программирование в Delphi. П Дарахвелидзе., Е. Марков- изд СПб и др· БХВ-Санкт-Петербург, 1999.
2. Программирование в Turbo-Pascal 7.0 и Delphi. Н.Б Культин.- 2изд. СПб и др: БХВ-Санкт-Петербург, 1999.
3. Delphi 7. С.И.Бобровский - изд. СПб и др. Питер: Inforcom press, 2004.
4. Delphi 6: Учебный курс. В.В. Фаронов -М.:Молгачева: Нолидж , 2001
5. Delphi 7 [Наиболее полное руководство]. А.Хомоненко, В.Гофман, Е.Мещеряков, В.Никифоров; под общей редакцией А.Хомоненко- СПб и др· БХВ-Санкт-Петербург, 2003