ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ»

ПРАКТИКУМ
ПО КУРСУ «ВИЗУАЛЬНЫЕ СРЕДЫ»

Составители:
И.Е. Воронина,
Н.В. Огаркова

Издательско-полиграфический центр
Воронежского государственного университета
2009

Утверждено научно-методическим советом факультета ПММ 26 декабря 2008 г., протокол № 4

Рецензент канд. физ.-мат. наук, доц. кафедры МО ЭВМ факультета ПММ О.Д. Горбенко

Пособие подготовлено на кафедре программного обеспечения и администрирования информационных систем факультета ПММ Воронежского государственного университета.

Рекомендуется для студентов 2 курса дневного отделения, обучающихся по специальности 010503 – Математическое обеспечение и администрирование информационных систем.

Для специальности 010503 – Математическое обеспечение и администрирование информационных систем

Содержание

Введение

Цель курса «Визуальные среды» – освоение современных инструментальных средств разработки программ.

Основные знания, умения и навыки, которыми студент должен овладеть в результате изучения дисциплины:

– понимание и использование объектно-ориентированного подхода, широко применяющегося в современных системах программирования;

– грамотное использование средств, уже имеющихся в среде разработки, таких, как типы данных, функции, модули, классы, формы и т. д.;

– освоение компонентного подхода к построению программ;

– приобретение навыков проектирования и реализации программ, управляемых событиями;

– умение проектировать и создавать современный пользовательский интерфейс.

Общие требования к лабораторным заданиям

1. Каждый проект должен храниться в отдельной папке.

2. Условие задачи должно храниться в отдельном текстовом документе, расположенном внутри папки проекта, либо должно быть сформулировано в комментариях в начале главного модуля проекта.

3. Все модули, в том числе и проект, должны быть переименованы в соответствии с их целевым назначением: недопустимо применение таких имен, как, например, `Project1`, `Unit1`, `Unit2` и т. д.

4. Имена всех компонентов также должны соответствовать их функциональному назначению: недопустимо использовать такие имена, как, например, `Form1`, `Edit1`, `Button6` и т. д.

5. Обязательно наличие комментариев ко всем составленным классам, типам, подпрограммам, блокам внутри подпрограмм и т. д.

6. Обязательно наличие удобного пользовательского интерфейса. Программа должна обеспечивать работу как с мышью, так и с клавиатурой. Следует обратить особое внимание на выбор компонентов при реализации интерфейса, а именно, выбор должен зависеть от вида входной/выходной информации, от диапазона допустимых значений, от того, насколько пользователю удобно с ним будет работать, и т. д.

7. Отображение информации на экране должно быть корректным. Например, если пользователь изменяет входные данные, то результат предыдущих действий следует автоматически либо убрать с экрана, либо сделать обработку с новыми входными данными.

Задание может быть не зачтено, если:

– программа не работает;

– программа не во всех случаях работает корректно;

– не соблюдены требования, предъявляемые к выполнению практического задания;

– интерфейс, предложенный в программе, нельзя считать оптимальным (можно создать более удобный интерфейс).

Структура практикума

В рамках практикума должны быть выполнены задания по *всем* представленным темам. Для каждой темы обозначена цель ее изучения, перечислены компоненты, которые необходимо освоить, предложен список заданий. Следует заметить, что для некоторых типов заданий сформулированы дополнительные требования, соблюдение которых также является обязательным.

Используемые обозначения:

«!» – компонент обязателен для изучения;

«*» – компонент желательно изучить.

Теоретический материал

Общая характеристика визуальных компонентов

Базовым для всех визуальных компонентов является класс `TControl`. Он обеспечивает основные функциональные атрибуты, такие как положение и размеры элемента, его заголовок, цвет и пр. В целом все визуальные компоненты можно разделить на две группы: оконные и неоконные элементы управления.

Оконный элемент управления представляет собой окно, предназначенное для решения конкретной задачи (кнопки, текстовые поля, полосы прокрутки). Базовым классом для оконных элементов управления является `TWinControl`. Оконные элементы управления содержат дескриптор окна (`window handler`). Дескриптором окна в операционной системе `Windows` называется 32-битная величина, однозначно определяющая данное окно. Приложение использует этот дескриптор для обращения к окну.

Для неоконных элементов управления базовым классом является `TGraphicControl`. Такие (неоконные) элементы управления не могут получать фокус ввода и быть родителями других компонентов. Достоинством неоконных элементов является то, что на них тратится меньше ресурсов (так как не нужен дескриптор окна). К таким компонентам относится, например, компонент `Label`.

Свойства

Свойства позволяют управлять внешним видом и поведением компонентов при проектировании и выполнении приложения. Свойства компонентов, доступные при проектировании приложения, также доступны и при его выполнении. Существуют свойства, которые доступны только во время работы приложения.

`Name` – (тип `TComponentName`) – каждый компонент, помещаемый на форму, получает имя по умолчанию (`Label1`, `Edit1`). Это имя затем может быть использовано программистом для управления компонентом во время выполнения программы. Чтобы упростить процесс написания программы и сделать ее более читабельной, на начальном этапе разработки приложения изменить имя на более осмысленное, соответствующее назначению компонента. Примеры имен компонентов приведены табл. 1.

Таблица 1. Возможные значения свойства Name

`From1`	`fmMain`	`FormMain`
`Edit1`	`edName`	`EditName`
`Button1`	`btnOk`	`ButtonOk`

`Action` – (тип `TBasicAction`) – используется для синхронизации работы нескольких взаимосвязанных элементов управления (см. `TActionList`).

`Align` – (тип `TAlign`) – способ выравнивания компонента внутри контейнера, например, внутри формы (`TForm`) или панели (`TPanel`). Значения, которые может принимать данное свойство, приведены ниже:

- `alNone` – так же, как и при проектировании приложения;
- `alTop` – по верхнему краю;
- `alBottom` – по нижнему краю;
- `alLeft` – по левому краю;
- `alRight` – по правому краю;
- `alClient` – по клиентской области;
- `alCustom` – зависит от родительского компонента.

`Caption` – (тип `TCaption`) – заголовок компонента. Один из символов заголовка может быть определен как «горячая» клавиша. Для этого перед символом необходимо указать знак «&». Например, если

```
ButtonOk.Caption:='&Ok'
```

то нажатие клавиш <Alt>+<O> будет эквивалентно нажатию на кнопку `ButtonOk`.

`Color` – (тип `TColor`) – цвет фона (поверхности) компонента.

`Constraints` – (тип `TSizeConstraints`) – ограничения на размер компонента. Это класс со своими подсвойствами, наиболее важными из которых являются: `MinHeight`, `MaxHeight`, `MinWidth`, `MaxWidth`.

`Ctl3D` – (тип `boolean`) – задает вид визуального компонента (`true` – объемный, `false` – плоский).

`Cursor` – (тип `TCursor`) – вид указателя мыши (например, `crDefault`, `crNone`, `crCross` и т. д.).

`Enabled` – (тип `boolean`) – определяет доступность компонента, т. е. способность компонента реагировать на поступающие сообщения (`true` – доступен, `false` – недоступен). Блокировка относится только к пользователю, а программно этот компонент, его свойства и методы доступны.

`Font` – (тип `TFont`) – определяет шрифт текста. Основные свойства класса TFont приведены в табл. 2.

Таблица 2. Основные свойства класса TFont

Name	TFontName	Название шрифта (Arial, Times New Roman, ...)
Size	Integer	Размер шрифта в пунктах (1 пункт = 1/72 дюйма)
Height	Integer	Размер шрифта в пикселах (если значение положительное, то межстрочный интервал учитывается, если отрицательное – не учитывается)
Style	TFontStyle	Стиль шрифта. Свойство множественного типа, которое может иметь следующие значения: – fsItalic (курсив), – fsBold (полужирный), – fsUnderline (подчеркнутый), – fsStrikeOut (перечеркнутый)
Color	TColor	Цвет шрифта

Handle – (тип HWND) – дескриптор окна. Используется при вызове API функций Windows.

Height и Width – (тип Integer) – вертикальный и горизонтальный размеры компонента в пикселах.

Left и Top – (тип Integer) – координаты левого верхнего угла компонента относительно содержащего его контейнера.

HelpContext – (тип THelpContext) – номер раздела справочной системы. (0 – не задан.)

Hint – (тип String) – текст всплывающей подсказки. При этом свойство ShowHint должно быть выставлено в значение true.

PopupMenu – (тип TPopupMenu) – локальное всплывающее меню, связанное с компонентом. При этом свойство AutoPopup должно быть выставлено в значение true.

Text – (тип TCaption) – содержит строку, связанную с компонентом (содержимое компонента).

TabOrder – (тип TTabOrder) – порядок получения компонентами контейнера фокуса при нажатии клавиши <Tab>.

TabStop – тип boolean) – возможность получения компонентом фокуса при нажатии клавиши <Tab>.

ReadOnly – (тип boolean) – разрешено ли связанному с вводом и редактированием текста элементу управления изменять находящийся в ней текст. (Запрет на редактирование относится только к пользователю, внутри программы текст можно менять.)

Parent – (тип TWinControl) – указывает на родительский элемент управления. Родительский компонент является контейнером для размещенных в нем других компонентов и отвечает за их прорисовку.

ParentColor, ParentCtl3D, ParentFont, ParentShowHint – (тип boolean) – показывают, будет ли компонент получать вышеуказанные значения от родителя (true – будет, false – не будет).

Visible – (тип boolean) – управляет видимостью компонента.

События

В языке Delphi события являются свойствами и принадлежат к соответствующему процедурному типу. Большинство событий носит нотификационный (уведомляющий) характер и имеет тип TNotifyEvent

```
TNotifyEvent = procedure (Sender : TObject) of object;
```

Sender для любого события указывает его источник.

При выборе элемента управления возникает событие OnClick типа TNotifyEvent. Это наиболее часто используемое событие.

При щелчке кнопкой мыши генерируются еще два события OnMouseDown и OnMouseUp типа TMouseEvent, при этом генерация событий происходит в следующем порядке:

1. OnMouseDown.
2. OnClick.
3. OnMouseUp.

```
TMouseEvent = procedure (Sender: TObject; Button: TMouseButton;
Shift: TShiftState; X, Y: Integer) of object;
```

- Button показывает, какая кнопка мыши нажата. Может принимать значения mbLeft, mbRight и mbMiddle;
- Shift определяет состояние клавиш <Alt>, <Ctrl>, <Shift> и кнопок мыши. Это параметр множественного типа. Допустимые значения приведены в ниже:
  - ssShift – нажата клавиша <Shift>;
  - ssAlt – нажата клавиша <Alt>;
  - ssCtrl – нажата клавиша <Ctrl>;
  - ssLeft – нажата левая кнопка мыши;
  - ssMiddle – нажата средняя кнопка мыши;
  - ssDouble – нажаты левая и правая кнопки мыши;
- X, Y – координаты курсора мыши.

При двойном щелчке левой кнопкой мыши генерируется событие OnDblClick типа TNotifyEvent, при этом события возникают в следующем порядке:

1. OnMouseDown
2. OnClick
3. OnMouseUp
4. OnDblCkick
5. OnMouseDown
6. OnMouseUp

При перемещении указателя мыши над визуальным компонентом *непрерывно* генерируется событие OnMouseMove типа TMouseMoveEvent:

```
TMouseMoveEvent = procedure (Sender: TObject; Shift: TShiftState;
X, Y: Integer) of object;
```

- Sender указывает, над каким элементом управления находится указатель мыши;
- X, Y – координаты указателя мыши относительно Sender;
- Shift – состояния клавиш <Alt>, <Ctrl>, <Shift> и кнопок мыши.

При вращении колеса мыши генерируются события OnMouseWheelDown, OnMouseWheelUp типа TMouseWheelUpDownEvent:

```
TMouseWheelUpDownEvent = procedure (Sender: TObject; Shift:
TShiftState; MousePos: TPoint; var Handled: Boolean) of object;
```

- Sender указывает источник события;

- Shift – состояния клавиш <Alt>, <Ctrl>, <Shift> и кнопок мыши;
- MousePos – координаты указателя мыши относительно Sender;
- Handler – будет ли визуальный компонент сам обрабатывать полученное сообщение (true) или передаст его родительскому компоненту (false).

Кроме того, при вращении колеса мыши также генерируется событие OnMouseWheel типа TMouseWheelEvent:

```
TMouseWheelEvent = procedure (Sender: TObject; Shift: TShiftState;
WheelDelta: Integer; MousePos: TPoint; var Handled: Boolean) of
object;
```

- Sender указывает источник события;
- Shift – состояния клавиш <Alt>, <Ctrl>, <Shift> и кнопок мыши;
- WheelDelta – показывает, на сколько условных единиц сдвинулось колесо мыши (если число положительное, то вперед, отрицательное – назад);
- MousePos – координаты указателя мыши относительно Sender;
- Handler – будет ли визуальный компонент сам обрабатывать полученное сообщение (true) или передаст его родительскому компоненту (false).

При работе с клавиатурой генерируются события OnKeyPress типа TKeyPressEvent и OnKeyDown, OnKeyUp типа TKeyEvent.

```
TKeyPressEvent = procedure (Sender: TObject; var Key: Char) of
object;
```

Здесь параметр Key содержит ASCII код нажатой клавиши, который может быть проанализирован и изменен. Если параметру Key присвоить значение #0 (символ с кодом 0), то это соответствует отмене нажатия клавиши.

```
TKeyEvent = procedure (Sender: TObject; var Key: Word; Shift:
TShiftState) of object;
```

Здесь параметр Key определяет клавишу (ее код), нажатую на клавиатуре. Для неалфавитно-цифровых клавиш можно использовать виртуальный код клавиши (Virtual Key codes) для того, чтобы определить, какая клавиша была нажата. Некоторые из этих значений приведены ниже.

- VK_LBUTTON – левая кнопка мыши;
- VK_RBUTTON – правая кнопка мыши;
- VK_CANCEL – Control+Break;
- VK_MBUTTON – средняя кнопка мыши;
- VK_BACK – клавиша <Backspace>;
- VK_TAB – клавиша <Tab>;
- VK_RETURN – клавиша <Enter>;
- VK_SHIFT – клавиша <Shift>;
- VK_CONTROL – клавиша <Ctrl>;
- VK_MENU – клавиша <Alt>.

Если событие OnKeyPress (OnKeyDown, OnKeyUp) необходимо реализовать для формы, то для того, чтобы форма раньше компонента получила и, как следствие, обработала это событие необходимо свойство формы KeyPreview выставить в значение true, а для того, чтобы событие не передавалось дочернему компоненту, необходимо внутри события выполнять присваивание вида Key := #0. Отдельные клавиши имеют особенности, например, при нажатии на клавишу <Tab> не возникают события OnKeyPress и OnKeyUp.

При получении оконным компонентом фокуса ввода возникает событие OnEnter типа TNotifyEvent, а при потере – OnExit типа TNotifyEvent.

Методы

Методы позволяют создавать и удалять объекты, прорисовывать их, отображать скрывать, а также выполнять другие операции.

SetFocus – устанавливает фокус ввода на оконный элемент управления. Если элемент не может получить фокус ввода, например он невидим, то возникает исключительная ситуация (ошибка времени выполнения).

CanFocus – проверяет, может ли элемент получить фокус ввода.

Clear – очистка содержимого компонентов, которые могут содержать текстовую информацию, например, ListBox, Memo, Edit.

Refresh – автоматически вызывает методы Invalidate и Update. Invalidate сообщает ОС Windows, что изображение требует прорисовки (при первой возможности ОС Windows выполнит эту операцию). Update вызывает немедленную прорисовку указанного объекта.

Perform предназначен для посылки сообщений оконным элементам управления.


```
function Perform (Msg: Cardinal; WParam, LParam: Longint): Longint;
```

Он посылает сообщение, текст которого задается параметром Msg; параметры WParam и LParam содержат дополнительную информацию о сообщениях.

Компоненты для ввода и отображения текста

Статический текст Label

Используется, когда необходимо отобразить текст, который не может быть отредактирован, например, используется в качестве заголовков компонентов, у которых нет свойства Caption.

Свойства

- Caption (string) – непосредственно текст;
- AutoSize (boolean) – автоматическая коррекция размеров;
- Alignment (TAlignment) – выравнивание текста внутри компонента. Допустимые значения представлены в табл. 3. Имеет смысл только в том случае, если свойство AutoSize имеет значение false.

Таблица 3. Допустимые значения свойства Alignment

Значение	Описание
taLeftJustify	По левому краю
taRightJustify	По правому краю
taCenter	По центру

- WordWarp (boolean) – автоматический перенос слов, если свойство имеет значение true. Имеет смысл только в том случае, если свойство AutoSize имеет значение false;
- Transparent (boolean) – показывает, прозрачна ли надпись, или имеет цвет Color;
- FocusControl – ассоциированный с надписью элемент управления;
- ShowAccelChar (boolean) – определяет, как интерпретируется символ «&» в Caption. Если свойство выставлено в значение true, то символ, следующий за знаком амперсанда, интерпретируется как горячая клавиша.

Строка редактирования Edit

Однострочный редактор для ввода и отображения текста.

Свойства

- Text (string) – сам текст;
- CharCase (TEditCharCase) – изменение регистра символов. Допустимые значения представлены в табл. 4.

Таблица 4. Допустимые значения свойства CharCase

Значение	Описание
ecNormal	Нормальное отображение символов
ecUpperCase	Преобразование символов в верхний регистр
ecLowerCase	Преобразование символов в нижний регистр

- MaxLength (integer) – максимальная длина вводимого текста;
- Modified (boolean) – признак, показывающий, было ли изменено свойство Text;
- PasswordChar (char) – символ-заполнитель, используемый при вводе пароля. Если свойство имеет значение #0, то символ-заполнитель не задан и отображаются вводимые символы, в противном случае отображается заданный символ;
- ReadOnly (boolean) – показывает, возможно ли изменение текста (true – невозможно);
- SelStart (integer) – позиция начала выделения фрагмента текста (нумерация начинается с нуля);
- SelLength (integer) – количество выделенных символов;
- SetText (string) – выделенный текст;
- AutoSelect (boolean) – если свойство имеет значение true, то при получении фокуса выделяется весь текст.

Методы

- SelectAll – выделить весь текст;
- ClearSelection – удаление выделенного текста;
- Clear – очистить весь текст;
- CopyToClipboard, CutToClipboard, PasteFromClipboard – методы для работы с буфером обмена.

Ограничение ввода MaskEdit

Ввод информации по шаблону.

- EditMask (string) – маска шаблона.

Однострочный редактор с надписью LabeledEdit

Свойства

- Editlabel (TBoundLabel) – собственно надпись, которая имеет свойства, рассмотренные для компонента Label, например, Caption, AutoSize и т. д.
- LabelPosition (TLabelPosition) – расположение надписи относительно поля редактирования. Допустимые значения свойства приведены в табл. 5.

Таблица 5. Допустимые значения свойства LabelPosition

Значение	Описание
lpAbove	Над текстом
lpBelow	Под текстом
lpLeft	Слева от текста
lpRight	Справа от текста

Класс TStrings

Это базовый класс для операций с набором (массивом) строк, имеющий наследников, например, класс TStringList. Его используют многие компоненты, например, Memo, ListBox, ComboBox, RadioGroup. Каждый элемент списка является строкой.

Свойства

- Strings [index] (string) доступ к строке текста по индексу (при этом следует учесть, что нумерация элементов начинается с нуля);
- Objects [index] (TObject) доступ к объектам, связанным со строками;
- Count (integer) – количество элементов в списке.

Методы

- Add, Insert – добавление/вставка строки;
- Delete – удаление строки по номеру (индексу);
- Clear – очистка списка;
- IndexOf – поиск элемента (строки) в списке. Возвращает значение номер строки, если она найдена, и –1 – в противном случае;
- LoadFromFile – загрузка списка строк из текстового файла;
- SaveToFile – сохранение списка строк в текстовый файл;
- Assign – копирование информации из списка строк, указанного в качестве параметра;
- Equals – сравнение на тождество исходного списка и списка строк, переданного в качестве параметра;
- AddStrings – добавление нескольких строк в список;
- AddObject – добавление в конец списка строки и связанного с ней объекта типа TObject.

Многострочный редактор Memo

- Text – предоставляет доступ ко всему содержимому;
- Lines (TStrings) – предоставляет возможность работы с отдельными строками;
- ScrollBars (TScrollStyle) – наличие полос прокрутки. Допустимые значения свойства приведены в табл. 6.

Таблица 6. Допустимые значения свойства ScrollBars

Значение	Описание
ssNone	Нет полос прокрутки
ssHorizontal	Только горизонтальная полоса прокрутки
ssVertical	Только вертикальная полоса прокрутки
ssBoth	Обе полосы прокрутки

- WantReturns (boolean) – реакция на нажатие клавиши <Enter>. Если свойство имеет значение true, то возможен ввод новой строки;
- WantTabs (boolean) – реакция на нажатие клавиши <Tab>.

Кроме этого, ряд свойств и методов компонента Memo эквивалентен свойствам и методам компонента Edit, например, Modified, Alignment, SelStart, CopyToClipboard и т. д.

Работа со списками

Основными видами списков, используемыми для проектирования пользовательского интерфейса, являются простой список (ListBox) и комбинированный список (ComboBox).

Общие свойства для обоих видов списков:

- Items (TStrings) – элементы списка;
- ItemIndex – выбранный (текущий) элемент списка (если элемент списка не выбран, то свойство имеет значение –1);
- MultySelect (boolean) – возможность выбора двух и более элементов;
- ExtentedSelect (boolean) – способ выбора нескольких элементов списка;
- SelCount (integer) – количество выбранных элементов списка;
- Selected [index] (boolean) – показывает, выбран ли элемент списка с номером index или нет;
- Sorted (boolean) – отсортирован ли список или нет. При выполнении присваивания Sorted := true выполняется сортировка элементов списка. После модификации списка его надо упорядочивать заново.

Простой список ListBox

- Columns (integer) – количество столбцов, которые одновременно видны. По умолчанию свойство имеет значение 0. Если не все пункты списка одновременно видны и свойство имеет значение, равное 1, то присутствует горизонтальная полоса прокрутки, иначе полоса прокрутки будет вертикальной;
- TopIndex (integer) – управление номером верхнего элемента.

Комбинированный список ComboBox

Представляет собой список и поле редактирования.

Свойства

- Style – определяет внешний вид и поведение комбинированного списка. Некоторые из значений приведены в табл. 7.

Таблица 7. Допустимые значения свойства Style

Значение	Описание
csDropDown	Раскрывающийся список с полем редактирования
csSimple	Поле редактирования с постоянно раскрытым списком
csDropDownList	Раскрывающийся список, который допускает только выбор элементов из списка

- DropDownCount (integer) – количество строк, которые одновременно отображаются в выпадающем списке;
- DroppedDown (boolean) – показывает, раскрыт ли список.

События

- OnDropDown – возникает, когда список раскрывается;
- OnCloseUp – возникает, когда список сворачивается;
- OnSelect – возникает, когда происходит выбор элемента из списка;
- OnChange – возникает, когда происходит изменение свойства Text.

Работа с кнопками

Стандартная кнопка Button

Свойства

- Caption (string) – надпись на кнопке;
- Default (boolean) – показывает, является ли кнопка кнопкой по умолчанию, т. е. реагирует ли она на клавиши <Space> и <Enter>;
- Cancel (boolean) – показывает, реагирует ли кнопка на нажатие клавиши <Esc>;
- ModalResult (TModalResult) – данное свойство используется для возврата модального результата, если кнопка используется для закрытия окна. Допустимые значения приведены в табл. 8.

Таблица 8. Допустимые значения свойства ModalResult

Значение	Описание
mrNone	Значение кнопки (результата) не определено, при этом окно не будет закрыто.
mrOk	Кнопка имеет значение OK
mrCancel	Кнопка имеет значение Cancel
mrAbort	Кнопка имеет значение Abort
mrRetry	Кнопка имеет значение Retry
mrIgnore	Кнопка имеет значение Ignore
mrYes	Кнопка имеет значение Yes
mrNo	Кнопка имеет значение No
mrAll	Кнопка имеет значение All
mrNoToAll	Кнопка имеет значение NoToAll
mrYesToAll	Кнопка имеет значение YesToAll

Кнопка с рисунком (BitBtn)

Такая кнопка, помимо надписи, может иметь рисунок и быть наделенной определенным смыслом (определено значение свойства ModalResult).

Свойства

- Glyph (TBitMap) – рисунок на кнопке. Если рисунок не определен, то свойство имеет значение None. Каждый рисунком может содержать до трех отдельных изображений (кнопка не нажата, недоступна, нажата);
- NumGlyph (integer) – количество изображений;
- Kind (TBitBtnKind) – один из предопредеденных видов для кнопки;
- Layout (TButtonLayout) – расположение изображения на поверхности. Допустимые значения приведены в табл. 9.

Таблица 9. Допустимые значения свойства Layout

Значение	Описание
blGlyphLeft	Слева от надписи
blGlyphRight	Справа от надписи
blGlyphTop	Над надписью
blGlyphBottom	Под надписью

- Margin (integer) – выравнивание изображения и текста относительно сторон кнопки. Если свойство имеет значение –1, то выравнивание идет по центру относительно расположения, в противном случае свойство показывает отступ в пикселах;
- Spacing (integer) – промежуток от изображения до текста в пикселах.

Кнопка быстрого доступа SpeedButton

Является неоконным элементом управления.

Свойства

- Down (boolean) – показывает, нажата ли кнопка или нет (находится ли она в нажатом состоянии);
- GroupIndex (integer) – определяет принадлежность кнопки определенной группе. Если свойство имеет значение –1, то это означает, что кнопка не относится ни к одной группе, в противном случае свойство показывает номер группы;
- AllowAllUp (boolean) – показывает, можно ли повторным щелчком отщелкнуть кнопку назад (true – можно).

Использование флажков и переключателей

Переключатель позволяет выбирать единственное значение из определенного множества значений.

Флажок – можно выставить несколько флажков одновременно.

Флажок CheckBox

Свойства

- Checked (boolean) – выбран ли элемент или нет;
- AllowGrayed (boolean) – если свойство имеет значение true, то для флажка допустимы три состояния (установлен, снят, недоступен), в противном случае только два (установлен и снят);
- State (TCheckBoxState) – анализ и установка одного из трех состояний флажка. Допустимые значения приведены в табл. 10.

Таблица 10. Допустимые значения свойства State

Значение	Описание
cbUnchecked	Снят
cbChecked	Установлен
cbGrayed	Недоступен

Переключатель RadioButton

Свойства

- Checked (boolean) – выбран ли элемент или нет (повторным щелчком нельзя снять пометку выбора).

Группа переключателей GroupBox

Свойства

- Items (TStrings) – надписи пунктов;
- ItemIndex (integer) – индекс выбранного пункта;
- Columns (integer) – число столбцов.

Задание 1

Тема

Создание элементарных GUI-приложений. Основные (простейшие) компоненты.

Цели

1. Научиться создавать простейшие GUI-приложения.
2. Научиться работать в интегрированной среде разработки `Delphi` с такими инструментами, как окно Инспектора объектов (`Object Inspector`) и окно Дерева объектов (`Object TreeView`).
3. Освоить свойства и события, общие (одинаковые) для многих компонентов, такие, как `Name`, `Caption`, `Visible`, `Enabled`, `OnClick`, `OnChange`, `OnEnter`, `OnExit` и др.
4. Научиться работать с такими компонентами, как `Edit` (!), LabelEdit (*), `MaskEdit` (*), `SpinEdit` (*), `Button` (!), `Bitbtn` (*), `SpeedButton` (*), `CheckBox` (!), `RadioButton` (!), `Label` (!) и др.
5. Научится реализовывать GUI-приложения для работы со строками, числами, логическими значениями (простыми типами данных).

Задания

1. В строке вводится последовательность целых чисел. Упорядочить эти числа по невозрастанию. *Требование*: если данные введены некорректно (есть буквы или знаки), то выдать соответствующее сообщение пользователю и выделить первый найденный некорректный символ или символы, если их несколько.

2. В строке вводится арифметическое выражение, состоящее из целых чисел и знаков «+» и «–». Вычислить значение этого выражения. *Требование*: во время ввода или модификации выражения нажатия на клавиши должны быть ограничены (если последней введена цифра, то за ней может быть нажата цифра, знак или пробел; если последним введенным символом является знак, то за ним могут быть введены либо пробел, либо цифры; для пробела определите ограничения самостоятельно).

3. В строке вводится последовательность русских слов, разделенных одним или несколькими пробелами. Упорядочить эти слова по алфавиту. *Требования*: предусмотреть, чтобы слова, содержащие буквы «Ё» или «ё», также правильно упорядочивались; при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (доступны только русские буквы и пробел).

4. В строке вводится последовательность слов, разделенных одним или несколькими пробелами. Под словом понимается последовательность латинских букв. Упорядочить эти слова по возрастанию их длины. *Требование*: если в строке не все слова удовлетворяют определению слова (например, между пробелами расположена последовательность символов, содержащая цифры), то выдать пользователю сообщение об этом.

5. Задана строка текста и некоторая подстрока-образец. Реализовать поиск подстроки в тексте. *Требования*: если результат поиска успешный, то найденная подстрока должна быть выделена, а надпись на кнопке «Найти» должна быть изменена на «Далее». При нажатии на кнопку «Далее» должен быть осуществлен поиск следующего вхождения заданной подстроки в текст. Если подстрока не найдена, то пользователю должно быть выдано сообщение об этом. Кнопка «Найти» должна быть доступна только в том случае, если задан и исходный текст и искомая подстрока.

6. Задана строка текста, содержащая слова, разделенные одним или несколькими пробелами. Под словом понимается последовательность латинских букв. Вывести те слова, перед которыми в последовательности находятся только предшествующие по алфавиту, а за ними – только последующие. *Требование*: при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (доступны только латинские буквы и пробел).

7. В строке вводится последовательность русских слов, разделенных одним или несколькими пробелами. Вывести эту же последовательность, но удалив из нее повторные вхождения слов. *Требования*: при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (доступны только русские буквы и пробел).

8. В строке вводится последовательность слов, разделенных одним или несколькими пробелами. Под словом понимается последовательность латинских букв. Вывести слова, которые встречаются в строке только один раз. *Требование*: если в строке не все слова удовлетворяют определению слова, например, между пробелами расположена последовательность символов, содержащая цифры, русские буквы или другие недопустимые символы, то выдать пользователю сообщение об этом.

9. В заданный непустой текст входят только буквы и цифры. Определить, удовлетворяет ли текст следующему условию: он начинается с некоторой непустой цифры, за которой следуют только латинские буквы, и их количество равно числовому значению этой цифры. *Требования*: при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (доступны только латинские буквы и цифры).

10. В заданный непустой текст входят только буквы и цифры. Определить, удовлетворяет ли он следующему условию: сумма значений цифр, входящих в текст, равна длине текста. *Требования*: при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (доступны только цифры и буквы).

11. Дан текст из строчных русских букв, за которым следует точка. Напечатать этот текст заглавными русскими буквами. *Требования*: при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (русские буквы и пробел доступны только до ввода точки, а после ее ввода ничего больше ввести нельзя).

12. Задана строка символов. Известно, что в ней могут встречаться цифры. Проверить, упорядочены ли числа, представленные цифрами, по убыванию. *Требование*: если в строке есть слова, которые не являются правильной записью целого числа (слова состоящие из букв и цифр), то выдать пользователю сообщение об этом.

13. Дан текст. Составить программу шифрации/дешифрации текста кодом Цезаря. Код Цезаря заключается в том, что задается алфавит в виде строки, а затем каждая буква введенного текста заменяется на ее номер в этом алфавите. Дешифрация – обратный процесс. Задается алфавит и строка чисел, которую надо расшифровать. Каждое число заменяется на букву, имеющую такой же порядковый номер в алфавите. Пример: «АБВГДЕЁЖЗИЙКЛМНО» – алфавит; «ЁЖИК» => «7 8 10 12», «ЛИМОН» => «13 10 14 16 15», «МАГ» => «14 1 4». *Требование*: если в строке, соответствующей алфавиту языка, есть повторяющиеся буквы, то выдать пользователю сообщение об ошибке.

14. Дан текст. Составить программу шифрации/дешифрации текста по следующему алгоритму: задаются два алфавита (обычный и для перекодировки, т. е. нешифрованный и шифрованный) в виде строк, а затем каждая буква введенного текста отыскивается в первом алфавите и заменяется на букву из второго алфавита с таким же номером, как и в первом алфавите. Дешифрация – обратный процесс. Задаются два алфавита и строка, которую надо расшифровать. Каждая буква зашифрованного текста отыскивается во втором алфавите и заменяется на букву из первого алфавита с таким же номером, как и во втором алфавите. Пример 1: «АБВГДЕЁЖЗИЙКЛМНО» – обычный алфавит, «абвгдеёжзийклмно» – зашифрованный алфавит; «ЁЖИК» => «ёжик», «ЛИМОН» => => «лимон», «МАГ» => «маг». Пример 2: «АБВГДЕЁЖЗИЙКЛМНО» – обычный алфавит, «~!@#$%^&*()_+=?/» – зашифрованный алфавит; «ЁЖИК» => => «^&(_», «ЛИМОН» => «+(=/?», «МАГ» => «=~#». *Требование*: при вводе первого алфавита необходимо контролировать, чтобы он не содержал повторяющихся символов, а при вводе второго – чтобы количество его букв совпадало с количеством букв первого алфавита. В случае ошибки выдать пользователю соответствующее сообщение.

15. Текст состоит из слов, разделенных одним или несколькими пробелами. Поменять местами слова в тексте по следующему принципу: первое – на последнее, второе – на предпоследнее и т. д.

16. Дана строка текста, которая состоит из слов, разделенных одним или несколькими пробелами. Распечатать эту строку таким образом, чтобы все слова располагались в центре (выравнивание по середине), длина строки осталась неизменной, а все лишние пробелы между словами были удалены.

17. Задан текст, состоящий из слов, разделенных одним или несколькими знаками «;». Найти слово в тексте, которое является обращением первого.

18. Дана непустая последовательность слов из латинских букв; между соседними словами – запятая, за последним словом – точка. Напечатать все буквы, которые входят в наибольшее количество слов последовательности. *Требование*: если в строке не все слова удовлетворяют определению слова, например, между запятыми расположена последовательность символов, содержащая цифры, русские буквы или другие недопустимые символы, то выдать пользователю сообщение об этом.

19. Задан текст, слова которого разделены пробелами, а за последним словом «@». Удалить из текста все слова, в которые входят символы, отличные от латинских букв. *Требования*: при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (любые символы доступны только до ввода символа «@», а после его ввода ничего больше ввести нельзя).

20. Задан текст, состоящий из слов, разделенных пробелами. В конце текста – точка. Слова состоят из латинских букв. Распечатать те слова текста, в которых нет повторяющихся букв. *Требования*: при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (доступны только латинские буквы, пробелы и точка). После ввода точки ничего больше ввести нельзя.

21. Определить, сколько различных латинских букв входит в заданный текст.

22. Напечатать заданное предложение, удаляя из него слова, целиком состоящие из вхождений не более чем двух букв.

23. Текст состоит из слов, разделенных одним или несколькими пробелами. Посчитать количество слов, содержащих ровно две буквы (буква задается пользователем). *Требования*: при вводе буквы предусмотреть, что пользователь не может ввести более одного символа, а если он вообще не задал такового, то выдать сообщение об этом.

24. Задано некоторое целое число в диапазоне от 2 до 16 – основание системы счисления (СС). Строка текста представляет собой последовательность чисел в заданной СС, разделенных знаками «+». Посчитать сумму этих чисел и вывести ее в заданной системе счисления. *Требования*: при вводе или модификации исходной строки предусмотреть ограничения нажатия клавиш (доступны только символы, соответствующие цифрам в заданной СС, пробел, знак «+»); при модификации основания СС исходная строка текста должна проверяться на допустимость и очищаться в случае необходимости; результат сложения также должен быть изменен (очищен) в случае изменения основания СС.

25. Задана строка текста, которая представляет собой запись римского числа. Проверить, является ли строка правильной записью римского числа, если да, то перевести его в арабское число, в противном случае вывести сообщение пользователю. *Требования*: при вводе или модификации исходной строки должны быть доступны только символы, соответствующие римским цифрам ('M', 'D', 'C', 'L', 'X', 'V', 'I').

Задание 2

Тема

Первое знакомство с уже реализованными в Delphi классами, умение их использовать (классы TStrings, TStringList). Компоненты, использующие классы TStrings и его потомки. Знакомство с другими компонентами.

Цели

1. Научиться использовать такие компоненты, как Memo (!), RichEdit (*), RagioGroup (!), ListBox (!), ComboBox (!) (одно из полей которых – класс TStrings).
2. Научиться использовать стандартные диалоговые окна, такие как OpenDialog (!), SaveDialog (!), ColorDialog (*), FontDailog (*) и др.
3. Научиться использовать меню: MainMenu (!), PopupMenu (*).

Требования

1. Программа должна позволять:
   a) создавать новый файл;
   b) открывать существующий;
   c) сохранять его;
   d) сохранять под другим именем;
   e) обрабатывать открытый файл;
   f) сохранять результат обработки (в случае необходимости);
   g) очищать результат обработки.
2. Группировка элементов управления по функциональности (использование компонентов Panel, GroupBox). Возможно использование компонентов SpeedButton, ToolBar, ImageList.
3. Поскольку программа предназначена для обработки текстовых файлов, то окна для их редактирования не должны быть маленькими. Кроме того, пользователь в некоторых случаях может самостоятельно устанавливать их размер. (Умение использовать свойства Align, Anckors, компонента Splitter.)

Задания

1. Дан текстовый файл, содержащий слова, разделенные одним или несколькими пробелами. Создать на его основе пять файлов. В первый файл должны войти слова длины пять, во второй – слова длины четыре и т. д. Если слов определенной длины не будет найдено, соответствующий файл должен быть пуст.

2. Дан текстовый файл, содержащий слова, разделенные одним или несколькими пробелами. Создать файл целых чисел, в котором каждой строке исходного файла соответствует в выходном файле число, равное количеству слов в ней. Пустой строке или строке, состоящей из одних пробелов, соответствует число ноль.

3. Дан текстовый файл, содержащий слова, разделенные одним или несколькими пробелами. Создать новый файл, каждая строка которого содержит слово исходного файла, имеющее номер $(n+1)\,div\,2$, где n – количество слов в соответствующей строке исходного файла.

4. Дан текстовый файл, содержащий слова, разделенные одним или несколькими пробелами. Создать файл, каждая строка которого должна содержать последнее из слов соответствующей строки исходного файла, в котором наибольшее число различных букв.

5. Дан текстовый файл, слова в котором состоят только из русских букв. Перенести в новый файл только те строки, которые предшествуют строкам, начинающимся с заданного слова.

6. Дан текстовый файл, слова в котором состоят только из русских букв. Переписать его содержимое в новый файл, сохраняя строчную структуру и удаляя пустые строки. Строка, состоящая только из одних пробелов, тоже считается пустой.

7. Дан текстовый файл. Упорядочить строки файла по количеству символов в них.

8. Дан текстовый файл, содержащий слова, разделенные одним или несколькими пробелами. Создать на основе новый файл, состоящий только из тех строк первого файла, которые содержат заданное слово.

9. Дан текстовый файл, слова в котором состоят только из русских букв. Создать на его основе новый файл, состоящий только из тех строк первого файла, которые не содержат заданное слово.

10. Дан текстовый файл. Известно, что в нем записаны целые числа. Записать в один файл все четные числа исходного файла, а в другой – все нечетные.

11. Даны два текстовых файла. Написать программу, которая сравнивала бы их на совпадение и выводила бы в качестве результата номер строки и номер символа, где встретилось первое отличие.

12. Дан текстовый файл. Получить новый файл, образованный из исходного заменой всех больших латинских и русских букв на малые.

13. Дан текстовый файл. Перенести в новый файл те строки исходного файла, которые начинаются и заканчиваются одной и той же буквой (вне зависимости от регистра).

14. Дан текстовый файл. Переписать его содержимое в файл, каждая строка которого имеет следующую структуру:

```
номер строки; строка; длина строки.
```

15. Дан текстовый файл, содержащий слова, разделенные одним или несколькими пробелами. Реализовать программу, позволяющую на основе исходного файла формировать новый файл, где каждое слово расположено на отдельной строке. Переход к новой строке в исходном файле соответствует пустой строке в новом файле.

16. Задан текстовый файл, состоящий из слов, расположенных на отдельной строке. Сформировать новый файл, который будет состоять из слов исходного файла, разделенных одним пробелом. Пустая строка исходного файла будет соответствовать переходу на новую строку в создаваемом файле.

17. Дан текстовый файл, слова в котором состоят только из русских букв. Создать новый файл, содержащий строки с наибольшим числом различных слов.

18. Дан текстовый файл, слова в котором состоят только из русских букв. Сформировать новый файл, каждая строка которого должна содержать слово наибольшей длины из соответствующей строки исходного файла. Если таких слов несколько, то они должны войти все.

19. Дан текстовый файл, слова в котором разделены одним или несколькими пробелами. Сформировать новый файл, удалив из исходного строки, заканчивающиеся заданным словом.

20. Дан текстовый файл, слова в котором разделены одним или несколькими пробелами. Сформировать три новых файла. Первый файл должен содержать только те строки исходного файла, количество слов в которых больше заданного значения k. Второй файл должен содержать только те строки исходного файла, количество слов в которых меньше заданного значения k, а третий – только строки, содержащие ровно k слов.

21. Дан текстовый файл, слова в котором состоят только из русских букв. Создать два выходных файла. В один переписать из каждой строки первые k слов, а в другой – оставшиеся. Если в строке меньше чем k слов, то во втором файле соответствующая строка должна быть пустой.

22. Дан текстовый файл. Сформировать на его основе новый файл, удалив из него те строки исходного файла, которые содержат наибольшее число различных символов, отличных от разделителя. Разделителем считается пробел, запятая, точка, двоеточие, точка с запятой, восклицательный и вопросительный знаки.

23. Дан текстовый файл, слова в котором разделены одним или несколькими пробелами. Перенести в новый файл только те строки, все слова которых состоят из повторяющихся букв.

24. Дан текстовый файл, слова в котором разделены одним или несколькими пробелами. Перенести в новый файл только те строки, в которых нет слов, состоящих из повторяющихся букв.

25. Дан текстовый файл, слова в котором состоят только из русских букв. Создать новый файл, исключив из строк исходного файла все слова, которые содержат хотя бы одну букву из заданного набора.

Задание 3

Тема

Работа с одномерными и двумерными массивами.

Цели

Научиться:

1) использовать такой компонент, как `StringGrid` (!);

2) использовать такие компоненты, как `TabControl` (*) и `PageControl` (*);

3) изменять свойства формы;

4) работать с несколькими формами.

Требования

1. Программа должна позволять:
   a) очищать данные (матрицу, массив);
   b) загружать данные из текстового файла;
   c) сохранять их;
   d) сохранять под другим именем;
   e) обрабатывать данные;
   f) сохранять результат обработки (в случае необходимости);
   g) очищать результат обработки;
   h) задавать (изменять) размер матрицы (массива).
2. В случае необходимости программа может содержать несколько простейших форм или использовать многостраничные элементы управления для организации удобного интерфейса.

Задания

1. Определить, является ли заданная квадратная матрица ортонормированной, т. е. такой, в которой скалярное произведение каждой пары различных строк равно 0, а скалярное произведение каждой строки на себя равно 1.

2. Дана целочисленная квадратная матрица порядка n. Найти номера строк, элементы которых образуют симметричные последовательности.

3. Задана целочисленная квадратная матрица. Преобразовать ее так, чтобы в начале каждой строки располагались отрицательные элементы, а в конце – положительные.

4. Задана квадратная матрица порядка n. Переставить ее элементы так, чтобы каждая строка была упорядочена по возрастанию, кроме того, элементы первого столбца тоже должны быть упорядочены по возрастанию.

5. Задана вещественная матрица $m \times n$. Обнулить те элементы, дробная часть которых не превосходит 0,1.

6. Задана квадратная матрица. Для каждого элемента матрицы вывести количество окружающих его нулей.

7. Задана квадратная матрица порядка n. Если количество отрицательных элементов матрицы больше n, то заменить отрицательные элементы их квадратами, кроме тех, которые расположены на главной диагонали. В противном случае заменить все положительные элементы средним арифметическим отрицательных элементов.

8. Задана матрица $m \times n$. Исключить из нее одинаковые строки.

9. Задана матрица $n \times n$. Подсчитать количество следующих пар: строка и столбец, которые при скалярном умножении дают число, меньшее среднего арифметического как данной строки, так и столбца.

10. Задана символьная матрица $m \times n$. Является ли она такой, что слово, образованное элементами каждой четной строки, является палиндромом, а слово, образованное элементами каждой нечетной строки, не содержит русских букв.

11. Задана вещественная квадратная матрица порядка n. Построить последовательность чисел $a_1, ..., a_n$ по следующему правилу: если в i-й строке матрицы элемент, принадлежащий главной диагонали, отрицателен, то a_i равно сумме положительных элементов i-й строки, в противном случае a_i равно произведению отрицательных элементов i-й строки.

12. Задана квадратная матрица порядка n. Если она симметрична относительно главной диагонали, то повернуть ее на 90° по часовой стрелке, иначе найти наименьший элемент из наибольших в каждой строке.

13. Задана квадратная матрица порядка n. Поменять местами главную диагональ со столбцом, содержащим наибольшее количество отрицательных элементов.

14. Дана целочисленная квадратная матрица порядка n. Найти номера тех строк, элементы каждой из которых одинаковы.

15. Дана целочисленная квадратная матрица порядка n. Выяснить, имеются ли в матрице ненулевые элементы, и если имеются, то указать индексы всех ненулевых элементов.

16. Дана вещественная квадратная матрица порядка n. Получить $x_1x_n + x_2x_{n-1} + ... + x_nx_1$, где x_i – наибольшее значение i-й строки матрицы.

17. Дана символьная неквадратная матрица. Найти номер последнего по порядку столбца, в котором находится наибольшее количество попарно различных символов.

18. Даны целые положительные числа n и m, вещественное число r, вещественная матрица $m \times n$. Получить значение $b_1r^{n-1} + b_2r^{n-2} + ... + b_n$, где b_k – первый по порядку положительный элемент в k-й строке матрицы ($k = 1, ..., n$); если в k-й строке нет положительных элементов, то $b_k = 0,5$.

19. Определить, является ли заданная целочисленная квадратная матрица магическим квадратом, т. е. такой, в которой суммы элементов во всех строках и столбцах одинаковы.

20. Составить программу, вычисляющую произведение двух матриц. Умножение, ввод и вывод матриц сделать с помощью подпрограмм.

21. Элемент матрицы называется локальным минимумом, если он строго меньше всех имеющихся у него соседей. Среди локальных минимумов заданной матрицы найти максимальный.

22. Матрица состоит из целых чисел, каждое их которых не меньше 5 и не больше 30. Две строки назовем похожими, если совпадают множества чисел, встречающихся в этих строках. Сколько строк матрицы похожи?

23. Даны две вещественные квадратные матрицы порядка n. Получить новую матрицу домножением каждой строки первой матрицы на наибольшее из значений элементов соответствующей строки второй матрицы.

24. Взаимно однозначное отображение элементов матрицы на себя можно задать с помощью двух целочисленных матриц: в первой указывать номер строки, куда переходит данный элемент, а во второй – номер столбца. Построить две матрицы, задающие отображение каждого элемента матрицы размера 10 × 10 на симметричный ему относительно главной диагонали.

25. Имеется таблица T результатов некоторого шахматного турнира, в котором участвовало n шахматистов ($n > 2$).

```
T : array [1..n,1..n] of (V, H, P, X),
```

где `T[i,j]=V`, если i-й участник выиграл у j-го (при этом `T[i,j]=P`), если i-й и j-й участники сыграли вничью, то `T[i,j]=H` и `T[i,i]=X`. Возможный вид таблицы (при $n = 3$):

$$\begin{pmatrix} X & V & P \\ P & X & H \\ V & H & X \end{pmatrix}.$$

За выигрыш дается 2 очка, за ничью – 1 очко, за проигрыш – 0 очков. Выдать на печать номера участников в порядке невозрастания набранных очков.

Задание 4

Тема

Создание SDI (MDI) GUI-приложения для работы с типизированными файлами.

Цели

1. Создание программы, имеющей стандартный интерфейс `Windows`.
2. Научиться синхронизировать элементы управления при помощи следующих компонентов: `ActionList`, `ActionManager`, `ActionMainMenuBar`, `ActionToolBar`.
3. Научиться работать с такими компонентами, как `ImageList`, `StatusBar`, `ProgressBar`, `ToolBar`, `CoolBar`.
4. Не использовать компоненты для хранения данных, поскольку они предназначены для отображения информации, а реализовывать для этого собственные внутренние структуры данных. Научиться создавать списки на основе стандартного класса `TList`.

Требования

1. Программа должна позволять:
   a) создавать новый файл;
   b) открывать существующий;
   c) сохранять его;
   d) сохранять под другим именем;
   e) загружать данные из текстового файла;
   f) сохранять данные в текстовый файл;
   g) добавлять данные, редактировать их, удалять;
   h) осуществлять поиск (выборку) с указанием количества найденных сведений по нескольким простым критериям;
   i) осуществлять сложный поиск (выборку), когда пользователь определяет совокупность полей, по которым будет идти поиск, и указывать значения для поиска.
2. Реализация класса, соответствующего одной записи. Реализация в нем (в случае необходимости) таких методов, как несколько конструкторов (пустой экземпляр или заполненный), загрузка данных из текстового файла, сохранение их в текстовый файл, отображение полей в класс `TStrings` (для соответствия одной строки `StrigGrid`) и других. Кроме этого, при реализации класса обязательно наличие секций, ограничивающих доступ к данным, а также свойств, обеспечивающих этот доступ.
3. Реализация класса «`Список из данных`», реализованного на базе стандартного класса `TList`, внутри которого должны быть такие методы, как загрузка данных из типизированного (текстового) файла, сохранение их в типизированный (текстовый) файл, отображение полей в класс `StringGrid`, различные методы поиска.
4. Программа должна удовлетворять всем требованиям Windows-приложения.

Задания

1. Рынок жилья

Поля	Тип данных	Рекомендуемые компоненты
Район	String, перечислимый	ComboBox
Адрес	String	Edit
Площадь	Real	MaskEdit, Edit
Количество комнат	Integer	SpinEdit, MaskEdit
Наличие балкона	Boolean	CheckBox, RadioGroup
Количество этажей	Integer	SpinEdit, MaskEdit
Наличие лифта	Boolean	CheckBox, RadioGroup
Цена	Real	MaskEdit, Edit

2. Магазин сотовых телефонов

Поле	Тип данных	Рекомендуемые компоненты
Марка	String	ComboBox, Edit
Модель	String	Edit. MaskEdit
Время работы в режиме ожидания	Integer	SpinEdit, MaskEdit, Edit
Время работы в режиме работы	Integer	SpinEdit, MaskEdit, Edit
Объем памяти	Integer	SpinEdit, MaskEdit, Edit
Полифония (количество голосов)	Integer	SpinEdit, MaskEdit, Edit
Тип дисплея (цветной или ч/б)	Boolean	CheckBox, RadioGroup
Наличие фотокамеры	Boolean	CheckBox, RadioGroup

3. Исторические события

Поле	Тип данных	Рекомендуемые компоненты
Название	String	Edit
Тип события	Перечислимый (война, битва, восстание и т. д.)	ComboBox, RadioGroup
Дата начала	TDateTime	DateTimePicker
Дата конца	TDateTime	DateTimePicker
Место	String	Edit
Основные действующие лица	Array [1..10] of string	Memo

4. Комнатные растения

Поле	Тип данных	Рекомендуемые компоненты
Название	String	Edit
Семейство	String	ComboBox, RadioGroup
Время цветения	Перечислимый (по месяцам)	ComboBox, RadioGroup
Количество соцветий	Integer	SpinEdit, Edit, MaskEdit
Название соцветий	Перечислимый	ComboBox, RadioGroup
Высота растения	Real	Edit, MaskEdit
Комнатное или нет	Boolean	CheckBox, RadioGroup

5. Животные

Поле	Тип данных	Рекомендуемые компоненты
Вид животного	Перечислимый	ComboBox, RadioGroup
Кличка	String	Edit
Время жизни	Integer	SpinEdit, Edit, MaskEdit
Хищник или травоядный	Boolean	CheckBox, RadioGroup
Место обитания	String	Edit
Опасен ли для людей	Boolean	CheckBox, RadioGroup

6. Меню кофейного автомата

Поле	Тип данных	Рекомендуемые компоненты
Название	String	Edit
Тип	Перечислимый (черный, капуччино и т. д.)	ComboBox, RadioGroup
Сахар	Integer (количество ложек)	SpinEdit, Edit, MaskEdit
Сливки	Boolean	CheckBox, RadioGroup
Цена	Real	Edit, MaskEdit

7. Библиотека

Поле	Тип данных	Рекомендуемые компоненты
Название	String	Edit
Автор	String	Edit, MaskEdit
Год издания	Integer	SpinEdit, Edit, MaskEdit
Издательство	String	ComboBox, Edit
Жанр	Перечислимый	ComboBox, RadioGroup
Есть ли в наличии	Boolean	CheckBox, RadioGroup

8. Отдел кадров

Поле	Тип данных	Рекомендуемые компоненты
ФИО	String	Edit
Дата рождения	TDateTime	DateTimePicker
Адрес	String	Edit
Телефон	String	MaskEdit, Edit
Должность	Перечислимый	ComboBox, RadioGroup
Стаж работы	Integer	SpinEdit, MaskEdit, Edit
Семейное положение	Перечислимый	ComboBox, RadioGroup

9. Микроволновые печи

Поле	Тип данных	Рекомендуемые компоненты
Фирма	String	ComboBox, Edit
Модель	String	Edit
Цена	Real	Edit, MaskEdit
Емкость	Integer	SpinEdit, MaskEdit, Edit
Наличие гриля	Boolean	CheckBox, ComboBox
Покрытие	Перечислимый	ComboBox. RadioGroup

10. Картотека видеофильмов

Поле	Тип данных	Рекомендуемые компоненты
Название фильма	String	Edit
Качество	Перечислимый	ComboBox. RadioGroup
Количество файлов	Integer	SpinEdit, MaskEdit, Edit
Кодек	String	ComboBox. Edit
Есть ли в наличие	Boolean	CheckBox, ComboBox
Кому отдал	String	Edit
Когда отдал	TDateTime	DateTimePicker

11. Магазин косметики

Поле	Тип данных	Рекомендуемые компоненты
Вид косметики	Перечислимый	RadioGroup
Фирма	TStringList	ComboBox
Дата выпуска	TDateTime	DateTimePicker,Calendar
Количество	Integer	SpinEdit
Цена	Real	Edit

12. Информация об автомобилях

Поле	Тип данных	Рекомендуемые компоненты
ФИО	String	Edit
Категория прав	Set of 'A'..'E'	CheckBox, CheckListBox
Модель автомобиля	String	ComboBox, RadioGroup
Номер автомобиля	String	MaskEdit, Edit
Номер двигателя	Integer	MaskEdit, Edit
Дата техосмотра	TDateTime	DateTimePicker, Calendar
Год выпуска авто	Integer	SpinEdit

13. Информация о прописке

Поле	Тип данных	Рекомендуемые компоненты
ФИО	String	LabeledEdit, Edit
Пол	String	RadioGroup, ComboBox
Улица	String	LabeledEdit, Edit
Номер дома	Integer	SpinEdit, MaskEdit
Дата	TDateTime	DataTimePicker

14. Музыкальный каталог

Поле	Тип данных	Рекомендуемые компоненты
Исполнитель	String	Edit
Песня	String	Edit
Группа	String	Edit
Альбом	String	Edit
Год выпуска	Диапазон 1..2006	SpinEdit
Жанр	Перечислимый	ComboBox, RadioGroup
Длительность	TDataTime	MaskEdit, SpinEdit

15. Информация о кроликах

Поле	Тип данных	Рекомендуемые компоненты
Кличка	String	Edit
Дата рождения	TDateTime	DateTimePicker, Calendar
Пол	Перечислимый	RadioGroup, ComboBox
Вес	Real	MaskEdit, Edit
Идентификационный номер	Integer	MaskEdit, SpinEdit

16. Информация о мониторах

Поле	Тип данных	Рекомендуемые компоненты
Вид монитора	Перечислимый	RadioGroup, ComboBox
Серийный номер	Integer	MaskEdit
Ширина	Real	MaskEdit, Edit
Высота	Real	MaskEdit, Edit
Есть в наличии или нет	Boolean	CheckBox, RadioGroup

17. Художественная галерея

Поле	Тип данных	Рекомендуемые компоненты
Автор	String	Edit, LabeledEdit
Название картины	String	Edit, LabeledEdit
Дата создания	Integer	SpinEdit, MaskEdit
Жанр картины	Перечислимый	RadioGroup, ComboBox
Цена	Real	Edit, MaskEdit
Продается или нет	Boolean	CheckBox

18. Клуб собаководов

Поле	Тип данных	Рекомендуемые компоненты
Порода	Перечислимый	RadioGroup, ComboBox
Возраст	Integer	SpinEdit, MaskEdit, Edit
Кличка	String	Edit, LabeledEdit
Владелец	String	Edit, LabeledEdit
Оценка	Перечислимый	RadioGroup, ComboBox
Допуск к разведению	Boolean	CheckBox

19. Информация о посетителях поликлиники

Поле	Тип данных	Рекомендуемые компоненты
ФИО	String	Edit, LabeledEdit
Дата рождения	TDateTime	DateTimePicker, Calendar
Адрес	String	Edit, LabeledEdit
Телефон	String	MaskEdit, Edit
Наличие страховки	Boolean	CheckBox, RadioGroup
Количество посещений	Integer	SpinEdit, MaskEdit, Edit

20. Информация о товаре на складе

Поле	Тип данных	Рекомендуемые компоненты
Код товара	Integer	Edit, MaskEdit
Вид товара	Перечислимый	ComboBox, RadioGroup
Наименование	String	Edit, LabeledEdit
Цена	Real	Edit, MaskEdit
Наличие на складе	Boolean	CheckBox, RadioGroup

21. Информация об игрушках в магазине

Поле	Тип данных	Рекомендуемые компоненты
Код	Integer	Edit, MaskEdit
Наименование	String	Edit, LabeledEdit
Возрастные границы	Два значения типа Integer	SpinEdit, Edit, MaskEdit
Цена	Real	Edit, MaskEdit
Удовлетворяет ли санитарным нормам	Boolean	CheckBox, RadioGroup

22. Информация о продуктах в магазине

Поле	Тип данных	Рекомендуемые компоненты
Наименование	String	Edit, LabeledEdit
Тип	Перечислимый	ComboBox, RadioGroup
Цена	Real	Edit, MaskEdit
Количество	Integer	SpinEdit, MaskEdit, Edit
Срок годности	TDateTime	DateTimePicker
Необходимость дозаказа	Boolean	CheckBox

23. Информация об учениках школы

Поле	Тип данных	Рекомендуемые компоненты
ФИО	String	Edit, LabeledEdit
Год обучения	Integer	SpinEdit, MaskEdit, Edit
Класс	Char	ComboBox, RadioGroup
Оценки за последнюю четверть	Array of byte	StringGrid
Группа здоровья	Перечислимый	RadioGroup, ComboBox

24. Информация о расписании

Поле	Тип данных	Рекомендуемые компоненты
Предмет	String	Edit, LabeledEdit
Преподаватель	String	Edit, LabeledEdit
День недели	Перечислимый	ComboBox, RadioGroup
Время	TDateTime	DateTimePicker
Аудитория	Integer	SpinEdit, MaskEdit
Вид занятия	Перечислимый	RadioGroup, ComboBox
Использование технических средств	Boolean	CheckBox, RadioGroup

25. Парикмахерская

Поле	Тип данных	Рекомендуемые компоненты
Название стрижки	String	Edit, LabeledEdit
Цена	Real	MaskEdit, Edit
С укладкой или без	Boolean	CheckBox, RaioGroup
Дополнительные средства	Множество на базе перечислимого типа	CheckListBox
Дата обслуживания	TDateTime	Calendar, DateTimePicker

Задание 5

Тема

Работа с графикой.

Цели

1. Научиться работать с такими классами, как `TCanvas`, `TGraphic`, `TPicture` и др.

2. Научиться работать с такими компонентами, как `Image` и `Timer`.

Задания

1. «Механические часы». На форме расположена графическая область с изображением часов, а также панель управления часами. С помощью панели управления пользователь может задать время, т. е. определить часы и минуты, а также установить время звонка. Кроме этого, он может завести часы, остановить их, завести будильник или остановить его завод, остановить звонок будильника. Часть тех же действий, а именно, установку часовой, минутной стрелки или стрелки завода, пользователь может сделать, используя графическую область, т. е. переместив стрелку с помощью мыши. При этом следует учесть, что данные на графической области и на панели управления должны быть синхронизированы, перемещение минутной стрелки должно влиять на расположение часовой, а перемещение часовой не должно влиять на расположение минутной.

2. «Погода». На форме расположена графическая область с изображением картины погодных условий, а также панель управления. К погодным условиям можно отнести следующие: солнечно, пасмурно, осадки. Если погода солнечная, то осадков быть не может, если пасмурная, то осадки могут как быть, так и не быть. В случае осадков, с неба (из облака) выпадают осадки, присущие данному времени года (например, летом не может быть снега, а зимой не может быть града). Кроме этого, следует учесть, что осадки могут быть совмещенными (снег с дождем или дождь с градом). На направление движения осадков (угол наклона падения) может влиять скорость и направление ветра. Программа должна работать в двух режимах: автоматическом и ручном. В автоматическом режиме смена времен года происходит автоматически через определенные промежутки времени, а в ручном пользователь должен иметь возможность самостоятельно задать все характеристики.

3. «Деревья (смена времен года)». На форме расположена графическая область с изображением деревьев, а также панель управления ими. Деревья могут быть лиственные и хвойные. Одной из разновидностей лиственных деревьев являются кусты (они ниже и все ветки растут от корня). На хвойных деревьях хвоя сохраняется в течение всего года, на лиственных деревьях листья сохраняются в зависимости от сезона. Весной все деревья подрастают, например, на половину расстояния до верха графической области. На лиственных деревь-

ях весной появляется листва, причем она имеет светло-зеленый цвет. В течение определенного времени (до осени) листья меняют свой цвет сначала до темно-зеленого, а затем становятся желтыми, после чего опадают. Зимой ничего не происходит, деревья спят. Используя панель управления, пользователь должен иметь возможность посадить любое дерево, срубить его, изменить время года. Программа должна работать в двух режимах: ручном и автоматическом. В ручном режиме пользователь сам меняет все параметры, а в автоматическом смена сезонов происходит через некоторые промежутки времени.

4. «Фрукты (смена сезонов)». На форме расположена графическая область с изображением дерева, а также панель управления. Весной дерево подрастает, например, на половину расстояния до верха графической области, на нем появляются листья. Затем на дереве расцветают цветы (не все одновременно), которые опадают в конце весны. Летом на месте цветов появляются плоды (яблоки или груши), которые также падают на землю в конце лета. Осенью листья желтеют и опадают. Зимой ничего не происходит, дерево спит. Используя панель управления, пользователь должен иметь возможность изменить время года, в соответствии с сезоном поместить на дереве либо цветок, либо плод (какой и где – зависит от пользователя), он также может дать команду, что цветок (плод) должен падать, причем какой именно – выбирает пользователь. Программа должна работать в двух режимах: ручном и автоматическом. В ручном режиме пользователь сам меняет все параметры, а в автоматическом смена сезонов происходит через некоторые промежутки времени.

5. «Цветочная клумба (смена сезонов)». На форме расположена графическая область с изображением клумбы, а также панель управления. Весной на газоне появляется трава, которая постепенно растет. Ближе к лету и все лето на газоне появляются и расцветают цветы, разные в зависимости от летнего месяца. Весной могут появиться, например, одуванчики, которые сначала желтые, а потом белые. Летом могут расцвести ромашки, колокольчики. Кроме того, на клумбе может расти земляника, на которой летом созревают плоды (сначала они растут зелеными, а потом краснеют). Клубника может пускать усы, из которых потом появляются новые кусты. Кроме того, на клумбе могут расти сорняки, которые не цветут и скорость роста у них больше. Используя панель управления, пользователь должен иметь возможность изменить месяц, в соответствии с ним посадить на клумбе растение, которое затем можно сорвать. Если растение – ягода, то можно не только сорвать растение, но и собрать плоды. Программа должна работать в двух режимах: автоматическом и ручном. В ручном режиме пользователь сам меняет все параметры, а в автоматическом смена месяцев происходит через некоторые промежутки времени.

6. «Человек, делающий зарядку». На форме расположена графическая область с изображением стоящего человека, а также панель управления. Человек делает зарядку под музыку (в графической области изображен магнитофон, который может либо молчать, либо из него доносится музыка). Человек может выполнять несколько видов упражнений: приседать, прыгать (со скакалкой),

выполнять махи руками и ногами. На панели управления должны быть расположены элементы управления, которые позволяют задать виды упражнений, менять скорость выполнения упражнения или останавливать выполнение упражнения.

7. «Морской бой». На форме расположена графическая область с изображением моря, неба, а также плывущего по морю корабля. Корабль плавает от левого края к правому и наоборот. В нижней части графической области расположена пушка, которая может поворачиваться. Из этой пушки можно сделать один или несколько выстрелов, причем снаряды, не достигшие цели (не поразившие цель и не достигшие линии горизонта), продолжают двигаться в заданном направлении. Если снаряд поразил цель, то количество очков играющего увеличивается. При поражении 10 кораблей подряд количество бонусных очков увеличивается, а скорость корабля возрастает. При этом корабль, начиная с некоторого количества очков, может изменять направление движения, не только достигнув края графической области, но и в любой другой момент времени. Игра заканчивается, если пользователь сделал три промаха подряд. На панели управления должны быть расположены кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее. Кроме того, на панели должны содержаться кнопки для управления пушкой.

8. «Битва с инопланетянами». На форме расположена графическая область с изображением космического пространства. В этом пространстве по направлению к вам двигаются летающие тарелки, которые в области видимости появляются случайным образом. В нижней части графической области расположен ваш корабль, который может двигаться только влево и вправо. Вы можете сделать один или несколько выстрелов, при этом снаряды, не достигшие цели (не поразившие цель и не достигшие конца видимого пространства), продолжают двигаться в заданном направлении. Если снаряд поразил цель, то количество очков играющего увеличивается. При поражении 10 кораблей подряд количество бонусных очков увеличивается, а скорость и количество летающих тарелок возрастает. Игра заканчивается, если хотя бы одна из летающих тарелок достигла вашего уровня. На панели управления должны быть расположены кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее. Кроме того, на панели должны содержаться кнопки управления вашим кораблем.

9. «Насекомые». На форме расположена графическая область с изображением поляны, а также панель управления, с помощью которой пользователь имеет возможность создавать новых насекомых, управлять их движением, а также удалять их. Насекомые должны быть нескольких видов, например, божья коровка, муха, паук и т. д. Такие насекомые, как паук и божья коровка, имеют лапки, которые при движении шевелятся, а муха, перемещаясь, машет крылышками. При движении насекомых следует отслеживать ситуации столкновения. «Выхода» насекомого за видимую область допускать нельзя.

10. «Чайник». На форме расположена графическая область с изображением газовой плиты и стоящего на ней чайника, а также панель управления ими. Панель управления должна предоставлять следующие возможности: включить (выключить) плиту, увеличить (уменьшить) пламя (при этом размер пламени должен меняться), налить (вылить) воду в (из) чайник(а) (при этом уровень воды должен быть виден: чайник прозрачный). Если плита включена, то вода нагревается, и в ней появляются маленькие пузырьки, которые всплывают наверх, потом вода закипает и пузырьки становятся крупнее. По мере кипения воды в чайнике ее уровень уменьшается. Если чайник остается пустым, а плита включена, то чайник начинает коптить.

11. «Построение графиков (диаграмм)». Программа должна предоставлять пользователю возможность построить диаграмму или график по некоторым данным. Диаграмма может быть либо круговой, либо столбчатой, а график может быть либо 2D, либо 3D. При этом пользователь должен иметь возможность менять такие параметры, как наличие осей, наличие подписей, цвет и т. д. (см. свойства компонента `Chart`). Кроме этого, программа должна позволять пользователю загружать данные из текстового файла, изменять их, сохранять в текстовом файле и др. Считается, что исходные данные заданы в декартовой системе координат.

12. «Лошадь». На форме расположена графическая область с изображением лошади, а также панель управления ею. Лошадь может ходить влево или вправо, прыгать, махать хвостом, вращать головой, а также есть траву. Программа должна позволять пользователю управлять движениями лошади как с панели управления, так и с клавиатуры. Кроме того, программа должна работать в автоматическом режиме, когда действия лошади определяются случайным образом. Пользователь должен иметь возможность в любой момент времени изменить режим работы программы.

13. «Змея». На форме расположена графическая область с изображением змеи, а также панель управления ею. Змея изображается подобно пружине. Она может ползать влево или вправо, махать хвостом, вращать головой, а также высовывать язык (шипеть). Программа должна позволять пользователю управлять движениями змеи как с панели управления, так и с клавиатуры. Кроме того, программа должна работать в автоматическом режиме, когда действия змеи определяются случайным образом. Пользователь должен иметь возможность в любой момент времени изменить режим работы программы.

14. «Яйцо – цыпленок – курица». Программа должна быть реализована в виде мультипликационного фильма. Сначала на экране появляется яйцо, из которого с течением времени появляется цыпленок. Цыпленок ходит по травке и клюет червяков. С каждым склеванным червяком цыпленок постепенно вырастает и превращается в курицу, которая сносит яйцо. Пользователь должен иметь возможность размещать на земле червяков.

15. «Бег с барьерами». На форме расположена графическая область с изображением беговой дорожки, на которой стоит спортсмен, а также панель

управления. Спортсмен может бежать и прыгать. По мере преодоления барьеров скорость появления барьеров увеличивается, а также меняется ритм их появления. За каждый «чисто» взятый барьер игроку начисляются очки. Игра завершается, когда спортсмен не смог преодолеть 10 барьеров. В программе должна быть таблица результатов с десятью лучшими участниками. Программа должна позволять пользователю управлять движениями спортсмена как с панели управления, так и с клавиатуры. На панели управления должны быть расположены кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее.

16. «Ловец яиц». На форме расположены графическая область и панель управления. В графической области изображены четыре насеста (слева вверху, слева внизу, справа внизу и справа вверху), по которым катятся яйца. В середине графической области расположен главный герой – ловец яиц (волк, Микки-Маус, человечек и т. д.), цель которого – собрать в корзинку наибольшее количество яиц. За каждое пойманное ловцом яйцо начисляется один балл, по мере набирания баллов скорость падения яиц и их частота увеличивается (но в каждый момент времени яйцо может появиться только на одном из четырех насестов). Иногда в графической области появляется фигурка (произвольная), которая отвлекает ловца, например, показывает ему язык. Ловцу по ходу игры разрешается разбить три яйца. Если яйцо падает на землю, когда отвлекающий объект есть на экране, то «снимается» пол-яйца, в противном случае – целое. Если пользователь набирает 1000 баллов, то все разбитые яйца восстанавливаются. Программа должна позволять пользователю управлять движениями ловца как с панели управления, так и с клавиатуры. На панели управления должны быть расположены кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее.

17. «Тетрис». Реализовать программу, которая является одним из вариантов игры «Тетрис»: падают разнообразные по форме фигуры, пользователь может перевернуть фигуру, сдвинуть ее влево, вправо или опустить вниз. Если после падения очередной фигуры ряд (ряды) заполнен полностью, то он уничтожается, а пользователю за это начисляются очки. По мере набирания очков скорость падения фигур должна увеличиваться. При этом пользователь (игрок) должен иметь возможность сам составлять падающие фигуры. Это делается до начала игры с помощью соответствующих настроек. Управление действиями (перевернуть фигуру, подвинуть ее вправо, влево или вниз) должно осуществляться как с панели инструментов, так и с клавиатуры. Кроме этого, на панели инструментов должны быть кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее.

18. «Бильярд». На форме расположена графическая область, представляющая собой бильярдный стол. На столе расположен один-единственный шар и кий, которым может управлять пользователь. Он может перемещать кий по экрану, поворачивать его, регулировать силу удара, выполнять удар. Пользователь может выполнять вышеперечисленные действия при помощи клавиатуры,

мыши, а также путем использования кнопок на панели управления. Движение шара после удара должно отвечать всем физическим законам (угол отражения от борта стола должен быть равен углу при столкновении с бортом, шар замедляет свое движение со временем и т. д.). После забивания шара пользователь должен иметь возможность либо выставить его на стол самостоятельно, либо это действие выполняет программа (случайным образом). Пользователь должен также иметь возможность перемещать шар по столу.

19. «Пенальти». На форме расположена графическая область, на которой изображены ворота в дальней ее части и мяч вблизи. Пользователь в начале игры может выбрать, за кого он будет играть: за вратаря или за выполняющего пенальти спортсмена. Роль второго игрока будет выполнять компьютер. В любом случае пользователь определяет область, в которую он будет бить (или которую будет защищать). Область, за которую отвечает компьютер, определяется случайным образом. Всего областей девять. После того как выполнится удар, в зависимости от результата, увеличиваются очки либо у игрока, либо у компьютера. По ходу игр выполняется заранее определенное количество ударов, например, пять. Выигрывает тот, кто забил больше голов. Пользователь может выполнять вышеперечисленные действия при помощи клавиатуры, мыши, а также при использовании кнопок на панели управления. Кроме того, на панели управления должны быть расположены кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее.

20. «Червяк». На экране расположена графическая область, по которой ползает червяк, а также панель управления движением червяка. Червяк может ползать влево, вправо, вверх и вниз. При этом червяк не может наступать на самого себя. Во время передвижения червяка на экране могут появляться различные фрукты. Червяк должен их поедать. Фрукты появляются лишь на некоторое время, а затем исчезают. Если червяк съедает фрукт, то он вырастает. И так каждый раз. При этом количество набранных игроком очков увеличивается. Игра прекращается, если червяк стал настолько большим, что при любом движении он наползает на себя. Пользователь может управлять движениями червяка как при помощи клавиатуры, так и при использовании кнопок на панели управления. Кроме того, на панели управления должны быть расположены кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее.

21. «Телефон». На форме расположена графическая область с телефонным аппаратом. У телефонного аппарата есть вращающийся диск для набора номера, а также трубка, которую пользователь может снять или положить. На форме также находится панель управления – аналог кнопочной модели телефона. Набор шестизначного телефонного номера можно осуществлять либо с помощью панели управления, либо используя цифровые клавиши клавиатуры, либо при помощи мыши (т. е. с помощью мыши «вращать» телефонный диск). Пользователь должен иметь возможность в любой момент осуществить «сброс» телефонного номера. При корректном наборе номера (любые шесть цифр) те-

лефон должен зазвонить. Необходимо предусмотреть возможность автоматического повторного набора последнего набранного номера (кнопка «Redial» у обычных телефонов).

22. «Тренировка памяти». В графической области, представленной на экране, случайным образом могут появляться и через некоторое время исчезать различные фигуры. Фигуры отличаются друг от друга как формой, так и цветом. При реализации следует учесть, что вся графическая область поделена на части, количество которых может изменяться от теста к тесту или задаваться пользователем. В каждой подобласти может появляться только одна фигура, форма и цвет которой определяется случайным образом. После того как фигуры из всех подобластей будут показаны пользователю, они все одновременно должны быть высвечены на экране. Пользователь должен указать, в каком порядке появлялись фигуры. При этом пользователь может указывать фигуры с помощью клавиатуры, мыши или кнопок на панели инструментов. При каждом правильном ответе пользователю начисляются очки. Если количество очков больше некоторого порогового значения, то пользователь имеет возможность для следующей тренировки (увеличивается количество фигур или уменьшается время их появления на экране). Кроме того, на панели управления должны быть расположены кнопки, позволяющие запустить тренировку, сделать паузу, продолжить тренировку, прекратить ее. Замечание: количество областей, пороговое значение, время, в течение которого фигура видна на экране, должны быть настраиваемыми.

23. «Мышь и кот». Реализовать компьютерную игру, в которой участвуют кот и мышь. Пользователь может выбрать роль либо кота, либо мыши. Смысл игры заключается в том, что кот стремится поймать мышь. Мышь может быть поймана, если она находится в поле зрения кота. В центре экрана изображена лишь голова кота. Глаза кота, отслеживая мышь, могут двигаться влево и вправо. Мышь, соответственно, перемещается влево и вправо в нижней части экрана. Если пользователь выбрал роль кота, то у него есть возможность управлять движением зрачков, а также дать команду «поймать». В этом случае роль мыши играет компьютер, который случайным образом определяет направление ее движения. Если пользователь выбрал роль мыши, то он может управлять только ее перемещением. Тогда компьютер будет играть роль кота, и мышь будет поймана только в том случае, если она находится в поле зрения кота больше порогового времени. Вне зависимости от выбранной роли пользователь может осуществлять действия как при помощи клавиатуры, так и при помощи панели управления. Если удалось поймать десять мышей подряд, то игра продолжается; предоставляется возможность поимки еще трех мышей. В противном случае игра прекращается. На панели управления должны быть расположены кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее.

24. «Обучение работе с клавиатурой». Нижняя часть графической области представляет собой фрагмент клавиатуры, отвечающей за набор текста. В верхней части области располагается строка, которую должен набрать пользо-

ватель во время тренировки. Для набора строки пользователю выделяется определенное количество времени. После того как дается команда начать тренировку, по строке начинает перемещаться курсор. Он указывает на ту букву, которую пользователь должен нажать. Тренировка заканчивается, если пользователь выполнил столько нажатий клавиш, сколько букв в тексте, либо если закончилось время. Далее определяется количество баллов, которое пользователь набрал во время выполнения задания. Каждая правильно набранная буква оценивается в один балл. Если набранное количество баллов выше порогового значения, то пользователь может тренироваться на новом уровне. Повышение уровня характеризуется тем, что текст для набора сложнее, а времени дается меньше. Все действия пользователя по нажатию клавиш синхронно отображаются на рисунке клавиатуры. Панель управления должна содержать кнопки, позволяющие начать или остановить тренировку. Кроме того, пользователю должна быть предоставлена возможность настроить тренировочные уровни: определить их количество, а для каждого уровня – текст и время выполнения.

25. «Жаба». На экране расположена графическая область. В нижней ее части изображено болото (пруд), на поверхности которого находятся три листа водного растения. Периодически один лист из трех может исчезать (уходить под воду). На одном листе сидит веселая жаба. Жаба может перепрыгивать с одного листа на другой. Прыжки могут быть короткими (на соседний лист) или длинными (через один). Жаба также может подпрыгивать вверх, чтобы съесть муху. Мухи появляются и исчезают случайным образом. Цель игры заключается в том, чтобы съесть как можно больше мух. При этом жаба не должна «утонуть» (лист, на котором она сидит, не должен уйти под воду). Пользователь может управлять жабой с клавиатуры или при помощи панели управления. На панели управления также должны быть расположены кнопки, позволяющие запустить игру, сделать паузу, продолжить игру, прекратить ее.

Список литературы

1. Delphi 7 / под общ. ред. А.Д. Хомоненко. – СПб. : БХВ-Петербург, 2003. – 1216 с.

2. Архангельский А.Я. Программирование в Delphi 6 / А.Я. Архангельский. – М. : БИНОМ, 2002. – 1120 с.

3. Архангельский А.Я. Программирование в Delphi 5 / А.Я. Архангельский. – М. : БИНОМ, 2000. – 1072 с.

4. Дарахвелидзе П.Г. Программирование в Delphi 5 / П.Г. Дарахвелидзе, Е.П. Марков, О.А. Котенок. – СПб. : БХВ-Санкт-Петербург, 2000. – 784 с.

5. Краснов М.В. OpenGL. Графика в проектах Delphi / М.В. Краснов. – СПб. : БХВ-Санкт-Петербург, 2000. – 352 с.

6. Краснов М.В. DirectX. Графика в проектах Delphi / М.В. Краснов. – СПб. : БХВ-Петербург, 2001. – 416 с.

7. Программирование на языке Паскаль : задачник / О.Ф. Ускова [и др.] ; под ред. О.Ф. Усковой. – СПб. : Питер, 2002. – 336 с.

Учебное издание

ПРАКТИКУМ ПО КУРСУ
«ВИЗУАЛЬНЫЕ СРЕДЫ»

Учебно-методическое пособие для вузов

Составители:
Воронина Ирина Евгеньевна,
Огаркова Наталья Владимировна

Подписано в печать 05.03.2009. Формат 60×84/16. Усл. печ. л. 2,7.
Тираж 50 экз. Заказ 299.

Издательско-полиграфический центр
Воронежского государственного университета.
394000, г. Воронеж, пл. им. Ленина, 10. Тел. (факс) +7 (4732) 598-026
http://www.ppc.vsu.ru; e-mail: pp_center@ppc.vsu.ru

Отпечатано в типографии Издательско-полиграфического центра
Воронежского государственного университета.
394000, г. Воронеж, ул. Пушкинская, 3