Автономная некоммерческая образовательная организация
высшего образования
«Воронежский экономико-правовой институт»
(АНОО ВО «ВЭПИ»)

АНОО ВО "ВОРОНЕЖСКИЙ ЭКОНОМИКО-ПРАВОВОЙ ИНСТИТУТ", АНОО ВО "ВЭПИ", ВЭПИ

Подписано цифровой подписью: АНОО ВО "ВОРОНЕЖСКИЙ ЭКОНОМИКО-ПРАВОВОЙ ИНСТИТУТ", АНОО ВО "ВЭПИ", ВЭПИ
Дата: 2021.01.01 14:07:27 +03'00'

УТВЕРЖДАЮ
Проректор
по учебно-методической работе
_____ А.Ю. Жильников
«__» _____ 2018 г.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

Б1.В.09 Разработка приложений на языке Delphi
(наименование дисциплины (модуля))

09.03.03 Прикладная информатика
(код и наименование направления подготовки)

Направленность (профиль) Прикладная информатика в экономике
(наименование направленности (профиля))

Квалификация выпускника Бакалавр
(наименование направленности (профиля))

Форма обучения Очная, заочная
(очная, заочная)

Рекомендованы к использованию Филиалами АНОО ВО «ВЭПИ»

Воронеж 2018

Методические рекомендации по выполнению лабораторных работ по дисциплине (модулю) рассмотрены и одобрены на заседании кафедры прикладной информатики.

Протокол от «13» декабря 2018 г. № 5

Заведующий кафедрой  Г.А. Курина

Разработчики:

Доцент  А. И. Кустов

ЛАБОРАТОРНЫЕ РАБОТЫ

Лабораторная работа № 1
«Основные понятия интегрированной среды разработки»

Цель работы: знать основные понятия интегрированной среды разработки.

1. Краткие теоретические сведения

Интегрированная среда разработки (IDE – Integrated Development Environment) – специальная программа, предоставляющая возможность удобной совместной работы с различными компонентами системы программирования.

Мы рассмотрели компоненты, входящие в систему программирования. Это и редакторы кода, и компиляторы, и сборщики, и отладчики, и многие другие. При первом же знакомстве со всеми этими программами становится понятно, что каждая из них может работать с разными начальными установками. Так, например, можно настроить множество параметров для редактора кода: цвет фона, цвет шрифта, шрифт, размер символа табуляции и еще сотню разных характеристик. Для компилятора можно указать, как оптимизировать код: по скорости, по размеру, никак не оптимизировать, а также есть возможность управления многими другими параметрами. Аналогично обстоит дело практически со всеми составляющими системы программирования.

Теперь представьте себе, как можно по очереди запускать все эти программы с огромным количеством разных параметров. То есть, сначала запускается редактор кода и пишется в нем программа. После этого подготовленная программа сохраняется, а затем закрывается редактор. Далее запускается компилятор, указав ему в качестве параметра файл с текстом программы и все необходимые настройки. Например, компилятор отработал и нашел 4 ошибки. Снова запускается текстовый редактор и в результате титанических усилий находятся эти строки и ошибки в них. После этого снова сохраняется программа, закрывается редактор и опять запускается компилятор. В результате компилятор создает объектный код, на этот раз без синтаксических ошибок. Теперь запускается сборщик, указывая ему кучу параметров и тот самый объектный файл. Если ошибок нет, то наконец, получится исполняемый файл, запускаете его и программа запустилась и «повисла». Или не повисла, но сказала, что уравнение не имеет корней, хотя точно известно, что решение есть. Это значит, что с семантикой что-то не то, иначе говоря, программа работает неправильно и ее необходимо отлаживать, искать ошибки. Для этого у нас есть отладчик.

Для устранения неудобств и повышения эффективности процесса разработки создатели систем программирования стали строить их в виде так называемых Интегрированных сред разработки. Термин «интегрированная» в

названии среды означает, что она включает в себя в качестве элементов все необходимые инструменты для выполнения полного цикла работ над программой: написания, компиляции, построения исполняемого модуля, запуска, отладки. Кроме того, интегрированные среды позволяют выполнять следующие операции:

1) визуально (в диалоге) производить быструю настройку параметров каждого из компонентов системы программирования;

2) сохранять разные системы настроек и загружать их по мере необходимости;

3) нажатием нескольких клавиш или выбором соответствующих пунктов меню осуществлять запуск одного или сразу нескольких компонентов системы программирования, автоматизируя процесс передачи им необходимых параметров.

2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

1. Описание основных понятий объектного языка программирования Delphi (Object Pascal).
2. Описание разделов Главного меню.
3. Описание Дизайнера форм.
4. Описание Инспектора объектов.
5. Описание Окна редактора исходного текста.

Содержание отчета:

1) цель работы;
2) задание на лабораторную работу для своего варианта;
3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1. Описание основных понятий объектного языка программирования Delphi (Object Pascal).
2. Описание Окна редактора исходного текста.

Лабораторная работа № 2
«Основные приемы размещения объектов на форме»

Цель работы: знать основные приемы размещения объектов на форме.

1. Краткие теоретические сведения

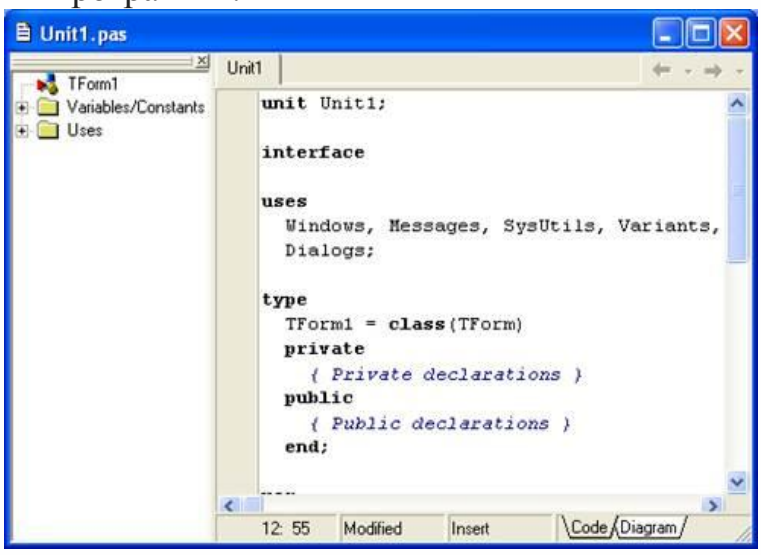
Среда программирования Delphi состоит из:

1. Главного меню. В нем располагаются стандартные операции работы с файлами (создать, сохранить, закрыть и т.д.)

Ниже располагаются вкладки объектов: *Standard*, *Additional* и прочие. Для открытия вкладки необходимо щелкнуть по ней кнопкой мыши.

2. Окно формы. На нее наносятся все объекты необходимые для работы создаваемого проекта.

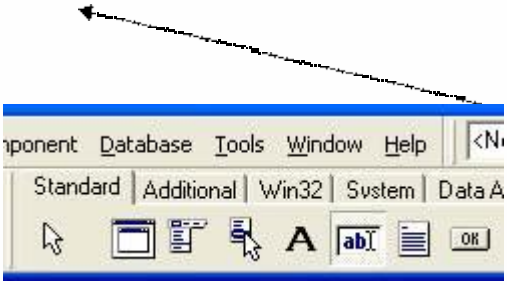
3. Окно программного кода. В нем записывается методы для работы программы.

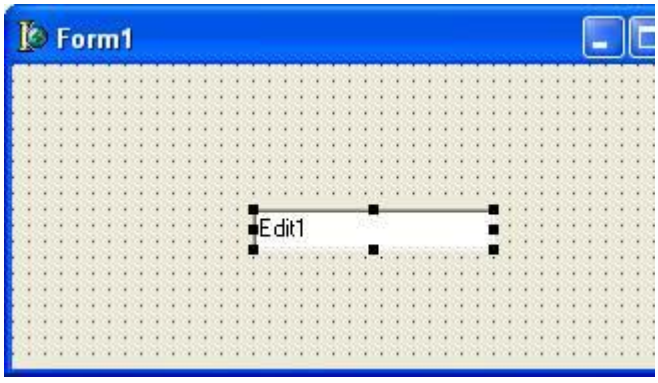


4. Окна инспектора объектов

Состоит из окна свойств (*Properties*) и событий (*Events*). При настройке свойств объектов на форме необходимо выделить данный объект щелчком мыши по нему и изменить поле слева от названия свойства. Например, для изменения свойства *Caption* (надпись) у объекта *Form1* (см. рис.) необходимо стереть в левом столбце *Form1*. Внимание! Нельзя путать свойства *Caption* и *Name*. Первое отвечает за название на форме, а второе – за имя объекта

Для нанесения объекта на форму нужно выбрать его в главном меню и щелкнуть в нужное место формы.





2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

1. Размещение кнопки на форме.
2. Изменение свойств объекта, размещенного на форме.
3. Анализ событий выбранного объекта.
4. Назначение обработчика событий заданному событию.
5. Компиляция и запуск проекта.

Содержание отчета:

1) цель работы;
2) задание на лабораторную работу для своего варианта;
3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1. Изменение свойств объекта, размещенного на форме.
2. Компиляция и запуск проекта.

Лабораторная работа № 3
«Структура проекта»

Цель работы: рассмотреть структуру проекта.

1. Краткие теоретические сведения

Проект– это разрабатываемое на языке программирования приложение.

Проект включает в себя не только форму с размещенными на ней компонентами, но и программные модули событийных процедур, которые описывают поведение объектов и взаимодействие объектов между собой.

Проект Delphi представляет собой набор программных единиц – модулей, которые хранятся в отдельных файлах.

Примечание. В Delphi существуют файл проекта и файлы проекта. Это разные вещи. Файл проекта – это главный файл проекта (головная программа), имеющий расширение .Dpr, файлы проекта – это набор всех файлов приложения.

Файл с расширением .Dpr (главный файл проекта) содержит основную информацию о проекте. По умолчанию этот файл называется Project1.dpr.

Файл с расширением. Pas– это файл программного модуля. В нем хранится текст программы на языке Object Pascal. Для каждой формы, входящей в состав проекта, создается отдельный программный модуль. По умолчанию эти файлы называются Unit1.pas,. Unit2.pas и т.д.

*Файл с расширением .Dfm*содержит информацию о внешнем виде формы. Этих файлов столько, сколько форм в проекте. Информация в них закодирована.

Файл с расширением .Res– это файл ресурсов проекта, в котором хранится информация о картинках, курсорах, иконках и т.п.

Файл с расширением .Exe – исполняемый файл приложения.

Файлы с расширениями .~Df, .~Pa– файлы со старыми версиями приложения.

У файлов с расширениями .Pas, .Dfm, .~Df, .~Pa всегда одинаковое имя (по умолчаниюUnit1).

У файлов с расширениями .Dpr, .Exe, .Res– также одинаковое имя (по умолчаниюProject1).

2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

1. Анализ структуры проекта.
2. Назначение и содержание модулей проекта.
3. Взаимодействие модулей.
4. Взаимодействие модулей и данных. Размещение модулей.

Содержание отчета:

1) цель работы;

2) задание на лабораторную работу для своего варианта;

3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1. Анализ структуры проекта.
2. Взаимодействие модулей и данных. Размещение модулей.

Лабораторная работа № 4
«Отладка проекта»

Цель работы: рассмотреть отладку проекта.

1. Краткие теоретические сведения

В Delphi имеется мощный встроенный отладчик, значительно упрощающий отладку программ. Основными инструментами отладки являются точки контрольного останова и окно наблюдения за переменными.

(1)Точки контрольного останова.

Точка контрольного останова определяет оператор в программе, перед выполнением которого программа прервет свою работу, и управление будет передано среде Delphi. Точка останова задается с помощью опции View|Debug windows|Breakponts.

Окно точек останова содержит список всех установленных в проекте точек, перед выполнением которых происходит прекращение работы программы и управление получает среда Delphi.

Для добавления новой точки следует щелкнуть по окну правой кнопкой мыши и выбрать опцию Add. В этом случае появляется окно, с помощью которого можно указать положение добавляемой точки:

FileName – определяет имя файла;

Line number – номер строки от начала файла (в момент появления окна оно содержит файл и строку с текстовым курсором);

Condition – можно указать условие останова в виде логического выражения (например, MyValue = Max-Value-12);

Pass count – количество проходов программы через контрольную точку без прерывания вычислений.

Окно точек останова (слева) и окно добавления новой точки (справа)

(2) Окно наблюдения.

Наблюдать за состоянием переменной или выражения можно с помощью специального окна, вызываемого опцией View|Debug windows|Watches.

Окно наблюдения используется в отладочном режиме для наблюдения за изменением значений выражений, помещенных в это окно.

Для добавления нового выражения следует щелкнуть по окну правой кнопкой мыши и выбрать опцию New Watch. В строке Expression ввести выражение. Окно Repeat count определяет количество показываемых элементов массивов данных; окно Digits указывает количество значащих цифр для отображения вещественных данных; переключатель Enabled разрешает или запрещает вычисление выражения. Остальные элементы определяют вид представления значения.

Значения переменных можно также посмотреть во время останова программы, наведя курсор мыши на переменную в тексте кода.

Окно наблюдения и окно добавления в него нового выражения

(3) Принудительное прерывание работы программы.

Если программа запущена из среды Delphi, ее работу можно прервать в любой момент с помощью клавиш Ctrl+F2, кнопки ESC, опцией Run|Program Pause или, наконец, установив точку контрольного останова в той части программы, которая выполняется в данный момент или будет выполнена.

(4) Трассировка программы.

Перед исполнением оператора, в котором установлена точка контрольного останова, работа программы будет прервана, управление получит среда Delphi, а в окне наблюдения отразится текущее значение наблюдаемых переменных и/или выражений.

Теперь программист может прослеживать работу программы по шагам с помощью клавиш F7 и F8 или инструментальных кнопок. При нажатии клавиши F8 будут выполнены запрограммированные в текущей строке действия, и работа программы прервется перед выполнением следующей строки текста программы.

Следует заметить, что контрольная точка останова выделяется по умолчанию красным цветом, а текущая прослеживаемая строка – синим. Если программа остановлена в контрольной точке, т. е. когда текущая строка совпадает со строкой останова, строка выделяется красным цветом. Признаком текущей строки является особое выделение строки в служебной зоне слева в окне редактора.

2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

1. Компиляция проекта.
2. Режимы компиляции.
3. Точки останова.
4. Переход между точками.
5. Просмотр текущего состояния переменных.
6. Прерывание выполнения программы.

Содержание отчета:

1) цель работы;
2) задание на лабораторную работу для своего варианта;
3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1. Просмотр текущего состояния переменных.
2. Прерывание выполнения программы.

Лабораторная работа № 5
«Настройка проекта»

Цель работы: узнать, как происходит настройка проекта.

1. Краткие теоретические сведения

Приложение собирается из многих элементов: форм, программных модулей, внешних библиотек, картинок, пиктограмм и др. Каждый элемент размещается в отдельном файле и имеет строго определенное назначение. Набор всех файлов, необходимых для создания приложения, называется проектом. Компилятор последовательно обрабатывает файлы проекта и строит из них выполняемый файл. Основные файлы проекта можно разделить на несколько типов: Проект имеет много различных параметров, с помощью которых вы управляете процессом компиляции и сборки приложения. Установить параметры проекта можно в окне **Project Options**. Для этого выберите в главном меню команду **Project / Options**... или в окне управления проектом вызовите контекстное меню элемента Project1 и выберите команду **Options**.... На экране появится диалоговое окно; вам останется лишь установить в нем нужные значения параметров.

Диалоговое окно параметров проекта состоит из нескольких вкладок. Параметров очень много, поэтому мы рассмотрим только те, которые используются наиболее часто.

На вкладке **Forms** можно задать главную форму приложения (**Main form**) и в списке **Auto-create forms** указать формы, которые будут создаваться одновременно с главной формой.

На вкладке **Application** можно задать название (**Title**) вашей программы. В среде Delphi дополнительно можно задать файл справки (**Help file**) и значок (**Icon**). На вкладке **Compiler** настраиваются параметры компилятора. Наиболее интересными из них являются переключатели **Optimization** (включает оптимизацию генерируемого кода) и **Use Debug DCUs** (позволяет отлаживать исходный код системных библиотек).

Оба этих переключателя полезны при отладке программы: первый следует выключить, а второй - включить.

На вкладке **Compiler Messages** настраивается чувствительность компилятора к подозрительному коду. Включив переключатели **Show hints** и **Show warnings**, вы будете получать от компилятора весьма полезные подсказки (hints) и предупреждения (warnings), а не только сообщения об ошибках.

2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

Параметры проекта.

1. Определение опций компилятора.
2. Определение путей модулей проекта.

3. Определение параметров редактора.
4. Определение параметров отладчика.

Содержание отчета:

1) цель работы;
2) задание на лабораторную работу для своего варианта;
3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1. Определение параметров редактора.
2. Определение параметров отладчика.

Лабораторная работа № 6
«Создание многооконных проектов»

Цель работы: научиться создавать многооконные проекты

1. Краткие теоретические сведения

Windows используются два вида приложений - это однодокументные и многодокументные. По названию можно догадаться, что в однодоментном приложении можно работать только с одним документом. Примером может быть хорошо известный windows блокнот. Да и почти все примеры на данном сайте тоже являются однодокументными приложениями.

Многодокументное приложение может использовать в своей работе несколько документов. Примером может быть Microsoft Word и Excel.

В многодокументном окне главная форма может содержать несколько дочерних окон. Свойство FormStyle определяет какой будет форма (главной или дочерней). Главная форма может быть только одна и свойство FormStyle должно быть fsMDIForm, а для дочерних форм - fsMDIChild.

В главной форме лучше не использовать такие средства управления как кнопки, таблицы строк и надписи, т.к. они будут мешать при просмотре дочерних форм. В интерфейс для главной формы лучше включать: меню, строку состояния, панель инструментов. Остальная незанятая область будет использована для дочерних форм.

Если при запуске приложения нам не требуется создание экземпляра дочернего окна, то его создание лучше организовать динамически, во время создания приложения.

Для лучшего понимания темы создадим приложение состоящее из 2-х форм. Первой присвоим, используя Инспектор Объектов, заголовок "Главная форма", имя - MainForm, FormStyle - fsMDIForm. Для второй "Дочерняя форма", ChaildForm, fsMDIChild - соответственно.

2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

1. Создание нескольких форм.
2. Переход между формами.
3. Переход между программными модулями.
4. Добавление и удаление форм из проекта.
5. Программный переход между формами.

Содержание отчета:

1) цель работы;
2) задание на лабораторную работу для своего варианта;
3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1. Создание нескольких форм.
2. Программный переход между формами.

Лабораторная работа № 7
«Защита информации в программах»

Цель работы: знать, как защитить информацию в программах.

1. Краткие теоретические сведения

Каталог готовых программных проектов в Delphi по информационной безопасности, которые могут обеспечить шифрование в Delphi файлов или текста. Такие программы предназначены для защиты секретной информации от посторонних глаз. При защите текстовых данных применяются алгоритмы, которые защищают открытый текст методом шифрования. Delphi 7 часто примется в учебных заведениях при написании таких программ по заданию от преподавателя. Также часто необходимо защищать целые файлы, обеспечивая шифрование файлов в Delphi с помощью специальных алгоритмов преобразования информации и вводом секретного ключа.

Процесс защиты данных представляется несколько задач:

- Шифрование - это процесс преобразования открытой исходной информации в закрытую с использованием ключа.
- Обратный процесс называется дешифрованием, то есть это процесс расшифровывания закрытой информации в открытую с помощью ключа. В данном разделе собраны готовые проекты в делфи по информационной безопасности для шифрования файлов и текстовой информации. Разработка программ защиты информации необходима для предотвращения утечек секретной информации, как у предприятия, так и обычного пользователя компьютера. Интересные программы на Delphi и Visual Basic с исходниками ориентированные на информационную безопасность, делая шифрование и расшифровывание данных.

2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

1. Создание парольной защиты.
2. Создание многопользовательского входа в программу.
3. Привязка паролей к пользователям.
4. Изменение статуса пользователей.

Привязка работы программы к категории пользователя.

Содержание отчета:

1) цель работы;
2) задание на лабораторную работу для своего варианта;
3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1) Создание парольной защиты.
2) Привязка работы программы к категории пользователя.

Лабораторная работа № 8
«Создание анимации в программе»

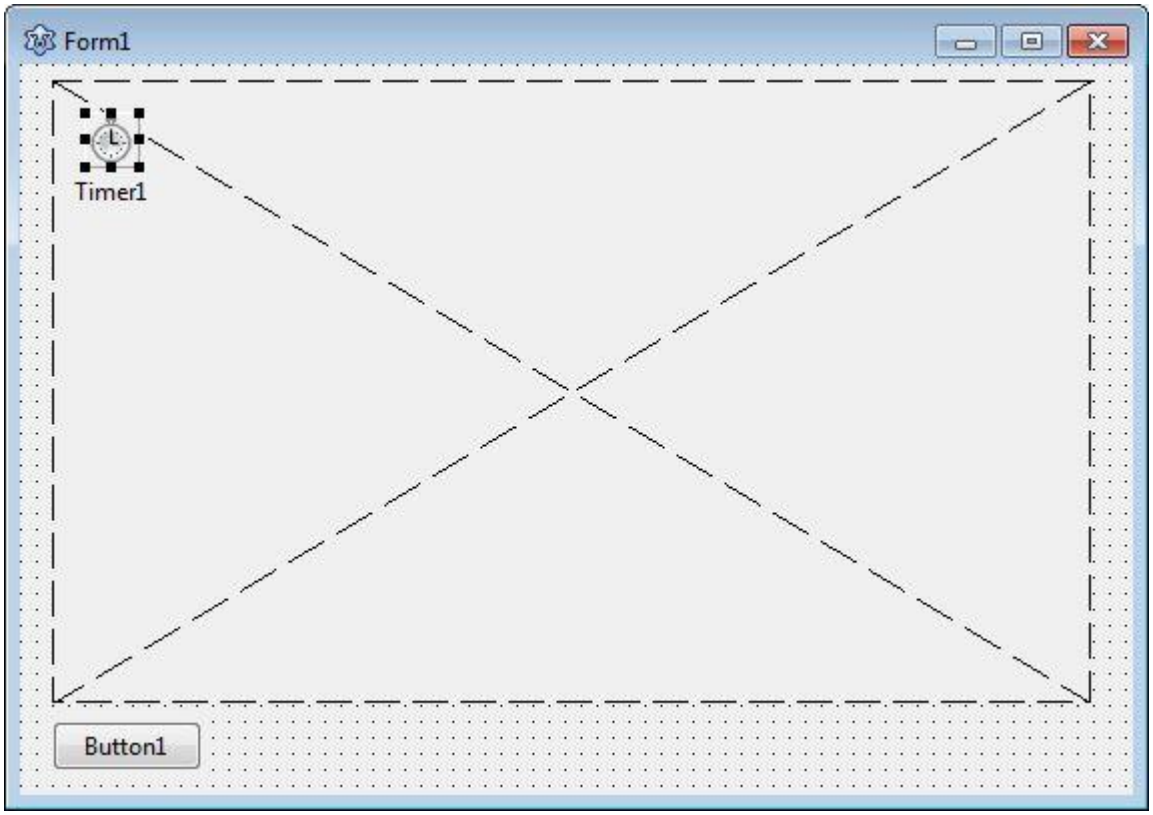
Цель работы: научиться создавать анимации в программе.

1. Краткие теоретические сведения

Линейное движение по однородному фону является довольно простым в плане программной реализации. Достаточно закрашивать объект цветом фона, изменять его координаты и прорисовывать в новом месте, повторяя эти действия через определенный интервал времени.

Для реализации анимации, помимо двух уже известных компонентов TPaintBox (поле для рисования) и TButton (кнопка запуска), понадобится компонент TTimer со вкладки System. Компонент Timer имеет единственное событие OnTimer, которое выполняется пока Timer включен с интервалом по времени, установленным в свойстве Interval.

Расположите компонент Timer1 на форме. Установите его свойства Timer1.Interval := 100 и Timer1.Enabled := false



В коде программы необходимо прописать три процедуры (см. урок "Процедуры и функции при построении изображений"). Процедуру отрисовки объекта procedure TForm1.Cloud, процедуру, отрабатывающую на событие OnTimer, - procedure TForm1.Timer1Timer и процедуру запуска анимации, срабатывающую на нажатие кнопки, procedure TForm1.Button1Click.

2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

1. Понятие таймера. Применение таймера.
2. Графика в программе.
3. Создание движения.
4. Управление движением в программе.

Содержание отчета:

1) цель работы;
2) задание на лабораторную работу для своего варианта;
3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1. Понятие таймера. Применение таймера.
2. Управление движением в программе.

Лабораторная работа № 9
«Разработка полнофункциональной программы»

Цель работы: рассмотреть процесс разработки полнофункциональной программы.

1. Краткие теоретические сведения

Создание приложения в среде Delphi можно условно разделить на несколько этапов:

1. Создание графического интерфейса будущего приложения

С помощью Панели инструментов на форму помещаются управляющие элементы, которые должны обеспечить взаимодействие приложения с пользователем.

2. Задание значений свойств объектов графического интерфейса

С помощью окна «Свойства объекта» задаются значения свойств управляющих элементов, помещенных ранее на форму.

3. Создание и редактирование программного кода

Для создания заготовки событийной процедуры необходимо осуществить двойной щелчок мышью по управляющему элементу. В окне «Редактор кода» появится заготовка событийной процедуры, имя которой состоит из двух частей: имени формы, содержащий управляющий элемент, и имени объекта и имени события (например,TForm1.Button1Click). Затем в окне «Редактор кода» производится ввод и редактирование программного кода процедуры.

4. Сохранение проекта

Т.к. проект включает в себя несколько файлов, рекомендуется для каждого проекта создать отдельную папку на диске. Сохранение проекта производится с помощью меню File:

1. Сначала необходимо сохранить форму и связанный с ней программный модуль (файл с расширением pas) с помощью команды Save As…. По умолчанию для файла формы предлагается имя Unit1.pas.

2. Далее необходимо сохранить файл главного модуля, который содержит описание проекта (файл с расширением dpr) с помощью команды Save Project As…

В процессе сохранения в папку проекта записываются вспомогательные файлы: файл с расширением res, описывающий ресурсы; файл с расширением dfm, описывающий форму, и некоторые другие файлы.

Сохраненный проект может выполняться только в самой системе программирования Delphi. Для того чтобы преобразовать проект в приложение, которое может выполняться непосредственно в среде операционной системы, необходимо сохранить проект в исполнимом файле (типа exe). Для компиляции проекта в исполнимый файл используется команда [Project-Compile].

2. Порядок выполнения работы и содержание отчета

Порядок выполнения работы:

1. Создание главного меню.
2. Создание системы подменю.
3. Создание информационной подсистемы.
4. Создание расчетной подсистемы.
5. Создание подсистемы формирования отчетов.

Содержание отчета:

1) цель работы;
2) задание на лабораторную работу для своего варианта;
3) алгоритм решаемого задания с необходимыми пояснениями;
4) выводы по работе.

3. Контрольные вопросы

1. Создание подсистемы формирования отчетов.
2. Создание информационной подсистемы.