Оглавление

Введение

Авторы книги давно хотели написать учебник по программированию, который был бы полезен пользователям различных операционных систем. Благодаря компании ALT Linux, мы попытались это сделать. В качестве языка программирования был выбран язык Free Pascal, который представляется нам ясным, логичным и гибким языком и приучает к хорошему стилю программирования. Свободно распространяемые компиляторы языка Free Pascal реализованы во многих дистрибутивах Linux, есть свободные компиляторы и для ОС Windows. Кроме того, в этой книге мы попытались познакомить читателя с принципами создания визуальных приложений в среде Lazarus.

В настоящее время существует множество подходов к изучению программирования. По мнению авторов, нельзя изучать программирование на каком-либо языке, не изучив методы разработки алгоритмов. Одним из наиболее наглядных методов составления алгоритмов является язык *блок-схем*. Об этом свидетельствует и многолетний опыт авторов преподавания программирования. Мы попытались написать учебник по алгоритмизации и программированию, насколько нам это удалось — судить читателю.

Авторы надеются, что читатель имеет первоначальные навыки работы на персональном компьютере под управлением ОС Linux или Windows и знаком со школьным курсом математики.

Книга состоит из десяти глав.

В *первой* главе читатель узнает о средствах разработки программ на Free Pascal, напишет свои первые программы.

Во *второй* главе изложены основные элементы языка (переменные, выражения, операторы) Free Pascal. Описаны простейшие операторы языка: присваивания и ввода-вывода, приведена структура программы, приведены примеры простейших программ линейной структуры.

Третья глава является одной из ключевых в изучении программирования. В ней изложена методика составления алгоритмов с помощью блок-схем. Приведено большое количество примеров блок-схем алгоритмов и программ различной сложности. Авторы рекомендуют внимательно разобрать все примеры и выполнить упражнения этой главы, и только после этого приступать к изучению последующих глав книги.

В *четвёртой* главе читатель на большом количестве примеров познакомится с подпрограммами. Описан механизм передачи параметров между подпрограммами. Один из параграфов посвящён рекурсивным подпрограммам. В завершении главы рассмотрен вопрос создания личных модулей.

Пятая и *шестая* главы посвящены изучению алгоритмов обработки массивов и матриц. Здесь же читатель познакомится и с реализацией этих алгоритмов на языке Free Pascal. Именно эти главы совместно с третьей являются ключом к пониманию принципов программирования.

Седьмая глава знакомит читателя с обработкой файлов на языке Free Pascal под управлением ОС Linux и Windows. На практических примерах изложен механизм прямого и последовательного доступа к файлам и обработки ошибок ввода-вывода. Описана работа с бестиповыми и текстовыми файлами.

Восьмая глава посвящена обработке строк и записей. Приведённые примеры позволят читателю разобраться с принципами обработки таблиц в языке Free Pascal.

В *девятой* главе авторы описали принципы объектно-ориентированного программирования и их реализацию в языке Free Pascal.

В *десятой* главе рассмотрены графические возможности Lazarus, изложено подробное описание алгоритма построения графиков непрерывных функций на экране дисплея. Приведены тексты программ изображения графиков функций с подробными комментариями.

К каждой теме прилагаются 25 вариантов задач для самостоятельного решения, что позволит использовать книгу не только начинающим самостоятельно изучать программирование, но и преподавателям в учебном процессе.

С рабочими материалами книги можно познакомиться на сайте Евгения Ростиславовича Алексеева — http://www.teacher.dn-ua.com.

Авторы благодарят компанию ALT Linux и лично Кирилла Маслинского за возможность издать эту книгу.

Авторы выражают благодарность своим родным за помощь и понимание.

Алексеев Е. Р., Чеснокова О. В., Кучер Т. В.

Донецк, январь 2009 г.

Глава 1

Средства разработки программ на языке Free Pascal

В этой главе мы начинаем знакомство с программированием на языке Free Pascal. *Язык программирования* Free Pascal ведёт своё начало от классического языка Pascal, который был разработан в конце 60-х годов XX века Никлаусом Виртом. Н. Вирт разрабатывал этот язык как учебный язык для своих студентов. С тех пор Pascal, сохранив простоту и структуру языка, разработанного Н. Виртом, превратился в мощное средство программирования. С помощью современного языка Pascal можно производить простые расчёты, разрабатывать программы для проведения сложных инженерных и экономических вычислений.

1.1 Процесс разработки программы

Разработку программы можно разбить на следующие этапы:

1) Составление алгоритма решения задачи. *Алгоритм* — это описание последовательности действий, которые необходимо выполнить для решения поставленной задачи.
2) Написание текста программы. *Текст программы* пишут на каком-либо языке программирования (например на Free Pascal) и вводят его в компьютер с помощью текстового редактора.
3) Отладка программы. *Отладка программы* — это процесс устранения ошибок из текста программы. Все ошибки делятся на синтаксические и логические. При наличии синтаксических ошибок (ошибок в написании операторов) программа не запускается. Подобные ошибки исправляются проще всего. Логические ошибки — это ошибки, при которых программа работает, но неправильно. В этом случае программа выдаёт не те результаты, которые ожидает разработчик или пользователь. Логические ошибки исправить сложнее, чем синтаксические, иногда для этого придётся переписывать отдельные участки программы, а иногда и перерабатывать весь алгоритм.

4) Тестирование программы. *Тестирование программы* — процесс выявления ошибок в работе программы.

Процессы отладки и тестирования сопровождаются неоднократным *запуском программы на выполнение*. Процесс запуска программы может быть осуществлён только после того, как введённая в компьютер программа на алгоритмическом языке Pascal[1] будет переведена в *двоичный машинный код* и создан *исполняемый файл*. Процесс перевода текста программы в машинный код называют *трансляцией*. Все трансляторы делятся на два класса:

- *интерпретаторы* — трансляторы, которые переводят каждый оператор программы в машинный код, и по мере перевода операторы выполняются процессором;
- *компиляторы* переводят всю программу целиком, и если перевод всей программы прошёл без ошибок, то полученный двоичный код можно запускать на выполнение.

Если в качестве транслятора выступает компилятор, то процесс перевода текста программы в машинный код называют *компиляцией*. При переводе программы с языка Pascal в машинный код используются именно компиляторы[2].

Рассмотрим основные этапы обработки компилятором программы на языке Pascal.

1) Компилятор анализирует, какие внешние библиотеки[3] нужно подключить, разбирает текст программы на составляющие элементы, проверяет синтаксические ошибки и в случае их отсутствия формирует объектный код (в Windows — файл с расширением **.obj**, в Linux — файл с расширением **.o**). Получаемый на этом этапе двоичный файл (объектный код) не включает в себя объектные коды подключаемых библиотек.
2) На втором этапе компоновщик подключает к объектному коду программы объектные коды библиотек и генерирует исполняемый код программы. Этот этап называется *компоновкой* или *сборкой программы*. Полученный на этом этапе исполняемый код программы можно запускать на выполнение.

На сегодняшний день существует множество компиляторов языка Pascal, среди которых можно выделить Borland Pascal, Delphi, а также свободно распространяемый кроссплатформенный компилятор языка Free Pascal и среду визуального программирования Lazarus.

1.2 Среда программирования Free Pascal

Рассмотрим *процесс установки компилятора* Free Pascal в ОС Linux. Для установки программ в операционной системе Linux служит менеджер пакетов. В разных дистрибутивах Linux используются различные менеджеры

[1] Как и на любом другом языке.

[2] Вместо термина «компилятор» в литературе иногда используют термин «транслятор компилирующего типа».

[3] В библиотеках языка Pascal хранится объектный (двоичный) код стандартных (таких, как $sin(x)$, $cos(x)$ и др.) функций и процедур языка.

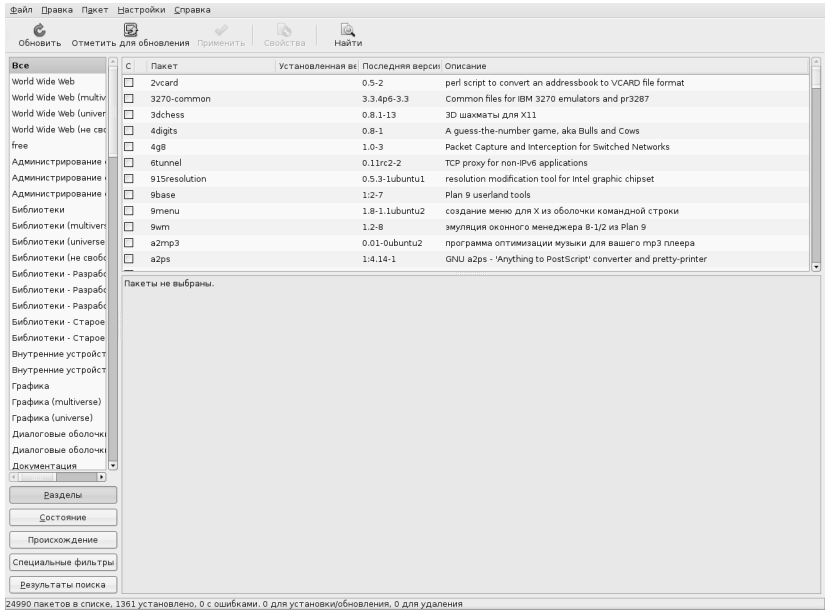

Рис. 1.1. Менеджер пакетов Synaptic

пакетов, например, в ALT Linux можно воспользоваться программой Synaptic. Окно Synaptic представлено на рис. 1.1. В школьной линейке дистрибутивов ALT Linux Free Pascal и Lazarus присутствуют сразу после установки операционной системы.

Обратите внимание, что для установки программ необходимо установить список источников программ (список репозиториев[4]).

Для установки Free Pascal в окне Synaptic (см. рис. 1.1) необходимо щёлкнуть по кнопке **Найти** и в открывшемся окне ввести **fpc** (см. рис. 1.2). Менеджер программ находит программу FreePascal, после чего в окне Synaptic необходимо отметить программы **fpc** (Free Pascal Compiler Meta Package) для установки (с помощью контекстного меню или с помощью кнопки **Отметить для обновления**) и начать установку, щёлкнув по кнопке **Применить**. После этого начнётся процесс загрузки пакетов из Интернета и их установки.

В состав метапакета **fpc** входит компилятор языка Free Pascal **fpc** и среда разработки **fp-ide**. Для запуска среды разработки в Linux необходимо просто в терминале набрать `fp`. На рис. 1.3 представлено окно среды разработки программ на языке Free Pascal в ОС Linux.

[4]Список репозиториев — список официальных сайтов, с которых можно устанавливать программы.

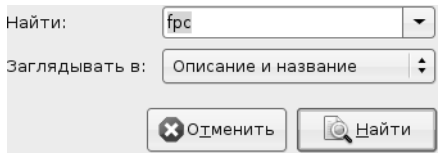

Рис. 1.2. Окно поиска компилятора Free Pascal в Synaptic

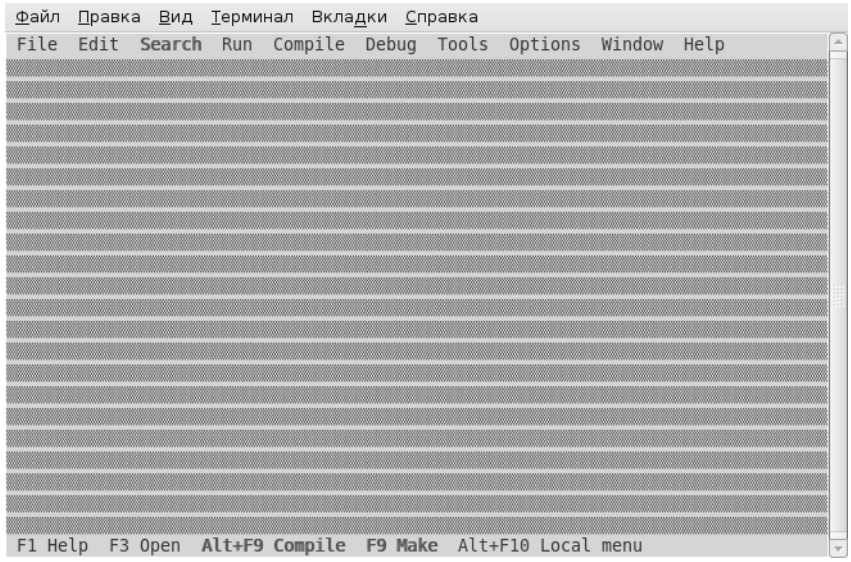

Рис. 1.3. Среда программирования Free Pascal в ОС Linux

Для установки Free Pascal в ОС Windows необходимо запустить скачанный со страницы загрузки[5] инсталяционный файл. Первое диалоговое окно сообщит о начале процесса установки Free Pascal на компьютер. Для продолжения установки во всех следующих окнах нужно выбирать кнопку **Next**, для возврата к предыдущему шагу — кнопку **Back**, а для прерывания процесса установки — кнопку **Cancel**. В следующем окне нужно определить путь для установки Free Pascal. По умолчанию установка происходит в корневой каталог диска C. Для выбора другого пути установки можно воспользоваться кнопкой **Browse...** Кроме того, в этом окне выводится информация о количестве свободного места на диске. В следующих четырёх окнах пользователь сможет выбрать из списка тип установки: **Full Installation** (полная), **Minimum Installation** (минимальная), **Custom Installation** (выбор компонентов), указать название устанавливаемого приложения в главном меню, выбрать типы файлов, поддерживаемых средой,

[5] `http://www.freepascal.org/down/i386/var.`

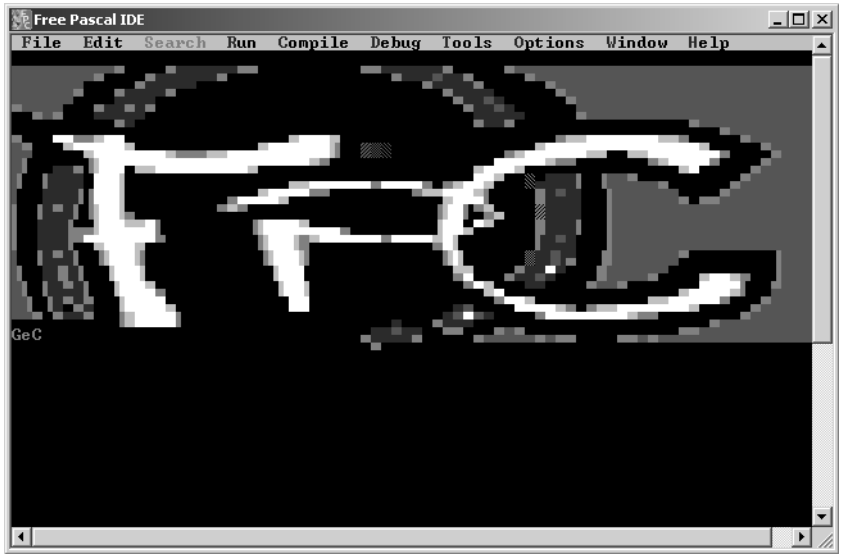

Рис. 1.4. Окно компилятора Free Pascal

и начать процесс установки Free Pascal, нажав кнопку **Install**. Контролировать процесс установки можно с помощью линейного индикатора.

Запуск среды программирования Free Pascal в Windows можно осуществить из главного меню: **Пуск** — **Программы** — **Free Pascal** — **Free Pascal**. На экране появится окно, представленное на рис. 1.4.

Установив пакет Free Pascal, мы получили компилятор и среду программирования.

Компилятор Free Pascal работает в командной строке. Для того чтобы создать исполняемый файл из текста программы, написанного на языке Pascal, необходимо выполнить команду

```
fpc name.pas
```

Здесь **fpc** — имя исполняемого файла компилятора командной строки Free Pascal, `name.pas` — имя файла с текстом программы. В результате в Linux будет создан исполняемый файл с именем `name` (в Windows имя исполняемого файла — **name.exe**).

При использовании компилятора **fpc** после компиляции автоматически происходит компоновка программы (запуск компоновщика **make**).

Технология работы с компилятором Free Pascal может быть такой: набираем текст программы в стандартном текстовом редакторе, затем в терминале запускаем компилятор, после исправления синтаксических ошибок запускаем исполняемый файл. При такой технологии работы с компилятором, необходимо

не забывать сохранять текст программы, иначе при запуске компилятора будет компилироваться старая версия текста программы.

Однако среда программирования позволяет значительно упростить процесс разработки программ. В состав среды программирования Free Pascal входит текстовый редактор, транслятор и отладчик. Рассмотрим их работу подробнее.

1.2.1 Работа в текстовом редакторе Free Pascal

С помощью редактора Free Pascal можно создавать и редактировать тексты программ. После открытия пустого окна (**File** — **New**) или загрузки текста программы (**File** — **Open**) мы находимся в *режиме редактирования*, что подтверждается наличием в окне *курсора* (небольшого мигающего прямоугольника). Для перехода из режима редактирования к главному меню нужно нажать клавишу **F10**, обратно — **Esc**. Кроме того, этот переход можно осуществить щелчком мыши либо по строке главного меню, либо по полю редактора.

Редактор Free Pascal обладает возможностями, характерными для большинства текстовых редакторов. Остановимся на некоторых особенностях.

Работа с фрагментами текста (блоками) в редакторе Free Pascal может осуществляться с помощью главного меню и функциональных клавиш. *Выделить фрагмент текста* можно с помощью клавиши **Shift** и клавиш перемещения курсора (стрелок).

В главном меню для работы с фрагментами текста предназначены команды *пункта редактирования* **Edit**:

- **Copy** (**Ctrl**+**C**) — копировать фрагмент в буфер;
- **Cut** (**Ctrl**+**X**) — вырезать фрагмент в буфер;
- **Paste** (**Ctrl**+**V**) — вставить фрагмент из буфера;
- **Clear** (**Ctrl**+**Del**) — очистить буфер;
- **Select All** — выделить весь текст в окне;
- **Unselect** — отменить выделение.

Команды **Copy** и **Cut** применяют только к выделенным фрагментам текста. Кроме того, пункт меню **Edit** содержит команды **Undo** и **Redo**, с помощью которых можно отменять и возвращать выполненные действия.

Комбинации клавиш, предназначенные для работы с блоком, таковы:

- **Ctrl**+**K**+**B** — пометить начало блока;
- **Ctrl**+**K**+**K** — пометить конец блока;
- **Ctrl**+**K**+**T** — пометить в качестве блока слово слева от курсора;
- **Ctrl**+**K**+**Y** — стереть блок;
- **Ctrl**+**K**+**C** — копировать блок в позицию, где находится курсор;
- **Ctrl**+**K**+**V** — переместить блок в позицию, где находится курсор;
- **Ctrl**+**K**+**W** — записать блок в файл;
- **Ctrl**+**K**+**R** — прочитать блок из файла;
- **Ctrl**+**K**+**P** — напечатать блок;
- **Ctrl**+**K**+**H** — снять пометку блока; повторное использование **Ctrl**+**K**+**H** вновь выделит блок.

Работа с файлами в среде Free Pascal осуществляется с помощью команд **File** главного меню и функциональных клавиш:

- **New** — открыть окно для создания новой программы;
- **Open** (**F3**) — открыть ранее созданный файл;
- **Save** (**F2**) — сохранить созданный файл;
- **Save As** — сохранить файл под другим именем;
- **Exit** (**Alt+X**) — выйти из среды программирования.

При *создании новой программы* ей по умолчанию присваивается стандартное имя `NONAMEOO.PAS` (NO NAME — нет имени).

При *первом сохранении файла* пользователю будет предложено ввести его имя. При *повторном сохранении* файл сохраняется под тем же именем. Команда **Save As** аналогична первому сохранению. Если файл не был сохранён, то при попытке завершить работу со средой, появится запрос о необходимости сохранить изменения в файле. При *открытии ранее созданного файла* его имя выбирают из списка существующих файлов.

В редакторе Free Pascal допускается *работа с несколькими окнами*. Переключаться между окнами можно двумя способами:

- для переключения в окно с номером от первого до девятого нажать комбинацию клавиш **Alt+i**, где **i** — номер окна (например **Alt+5** — вызов пятого окна);
- для вывода списка окон на экран нажать комбинацию клавиш **Alt+0**, появится *список активных окон*, в котором будет необходимо выбрать нужное и нажать **Enter**.

1.2.2 Запуск программы в среде Free Pascal и просмотр результатов

После того как текст программы набран, его следует перевести в машинный код. Для этого необходимо *вызвать транслятор* с помощью команды **Compile** — **Compile** (комбинация клавиш **Alt+F9**). На первом этапе транслятор проверяет наличие синтаксических ошибок. Если в программе нет синтаксических ошибок, то на экране сообщается о количестве строк транслированной программы и объёме доступной оперативной памяти.

Если на каком-либо этапе транслятор обнаружит ошибку, то в окне редактора курсор укажет ту строку программы, в которой ошибка была обнаружена. При этом в верхней строке редактора появится краткое диагностическое сообщение о причине ошибки.

Для *запуска транслированной программы* необходимо выполнить команду **Run** — **Run** (комбинация клавиш **Ctrl+F9**), после чего на экране появляется окно командной строки, в котором пользователь и осуществляет диалог с программой. После завершения работы программы вновь появляется экран среды Free Pascal.

Для *просмотра результатов работы программы* в ОС Windows необходимо нажать комбинацию клавиш **Alt+F5**. Для возврата в оболочку следует нажать любую клавишу.

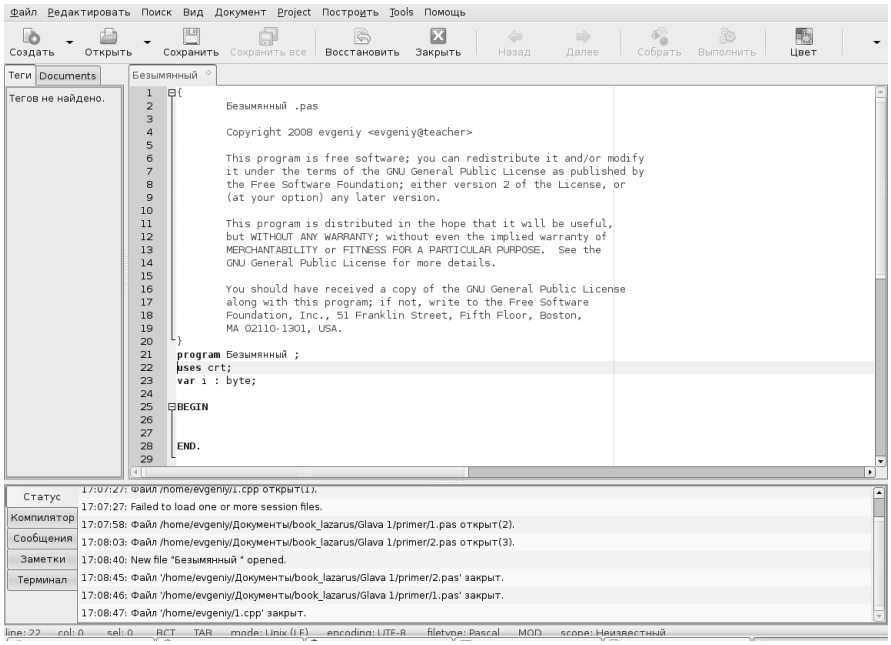

Рис. 1.5. Окно Geany с шаблоном программы на Free Pascal

Однако при использовании среды программирования существует проблема вывода кириллического текста в консольном приложении.

Альтернативой среды программирования Free Pascal является текстовый редактор Geany (http://www.geany.org), который позволяет разрабатывать программы на различных языках программирования.

1.3 Среда разработки Geany

Среда разработки (её также иногда называют текстовым редактором) Geany есть в репозитории большинства современных дистрибутивов Linux[6]. Разработка программ с использованием Geany довольна удобна. Geany можно установить стандартным для вашего дистрибутива способом, например, с помощью менеджера пакетов Synaptic.

Последовательно рассмотрим основные этапы разработки программы с использованием Geany.

1) Необходимо создать шаблон приложения на Pascal (или другом языке программирования) с помощью команды **Файл — New (with Template) — Pascal source file**. После чего появится окно с шаблоном исходного кода

[6]Существует версия Geany и для Windows (`http://www.geany.org/Download/Releases`).

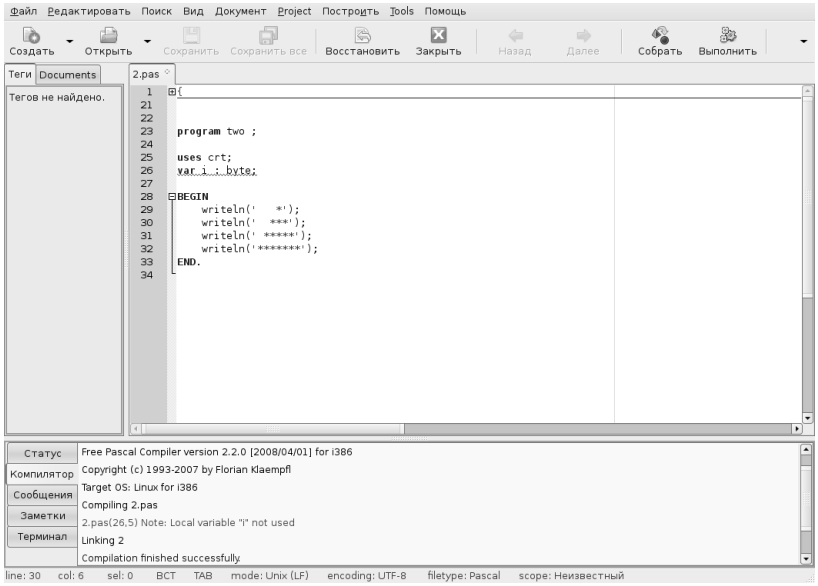

Рис. 1.6. Окно Geany с текстом программы на языке Free Pascal

(см. рис. 1.5), в котором необходимо ввести текст программы и сохранить его (см. рис. 1.6).

2) Для компиляции и запуска программы на выполнение служит пункт меню **Построить**. Для компиляции программы следует использовать команду **Построить** — **Собрать** (**F8**). В этом случае будет создан файл с объектным кодом программы и исполняемый файл. После компиляции программы ниже окна с программой будет представлен подробный отчёт (см. рис. 1.6) о результатах компиляции. Этот отчёт следует внимательно изучить, поскольку он позволяет быстро исправлять синтаксические ошибки. В сообщении об ошибке указана строка, где она найдена, и её краткое описание на английском языке. В отчётах по результатам компиляции могут быть *ошибки* (*error*) и *сообщения* (*warning*). Сообщения — это обнаруженные компилятором неточности, при которых, однако, возможно создание исполняемого кода программы.

3) Для запуска программы следует выполнить команду **Построить** — **Выполнить** (**F5**). После чего на экране появится окно терминала (см. рис. 1.7), в котором можно вводить данные и увидеть результаты работы программы.

В Geany можно настроить команды вызова компиляции, компоновки и запуска. Для этого служит команда **Построить** — **Установить включения и аргументы**. Это окно для работы с файлами с расширением `pas` представлено на

```
   *
  ***
 *****
*******
```

Рис. 1.7. Окно терминала с результатами работы программы

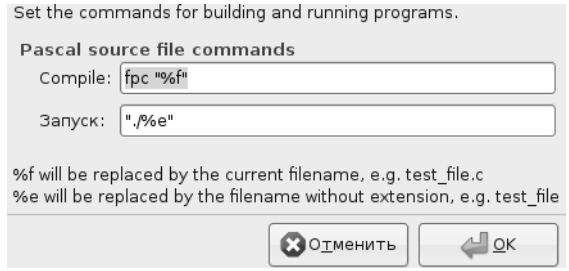

Рис. 1.8. Окно Установить включения и аргументы для Free Pascal

рис. 1.8. При настройке строк **Compile** и **Запуск** следует учитывать, что **%f** — имя компилируемого файла, **%e** — имя файла без расширения.

Выбор среды для разработки консольных программ на Free Pascal — дело пользователя. Авторы советуют читателю под управлением ОС Linux использовать Geany[7], хотя для набора текста программы можно использовать и обычный текстовый редактор (например **gedit**, **tea**, **kate** и др.), а компиляцию осуществлять в терминале. Под управлением Windows логичнее использовать **fp-ide**.

1.4 Среда визуального программирования Lazarus

Lazarus — это *среда визуального программирования*. Здесь программист получает возможность не просто создавать программный код, но и наглядно (визуально) показывать системе, что бы он хотел увидеть.

Технология визуального программирования позволяет строить интерфейс[8] будущей программы из специальных компонентов, реализующих нужные свойства. Количество таких компонентов достаточно велико. Каждый компонент содержит готовый программный код и все необходимые для работы данные, что избавляет программиста от создания того, что уже создано ранее. Подобный подход во много раз сокращает время написания программы. Кроме того, быстрота

[7]Это субъективный совет авторов.
[8]Интерфейс — диалог, обмен информацией.

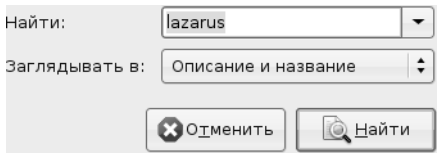

Рис. 1.9. Окно поиска пакета Lazarus для установки

создания программного кода в Lazarus достигается за счёт того, что значительная часть текста формируется автоматически.

Среда визуального программирования Lazarus сочетает в себе компилятор, объектно-ориентированные средства визуального программирования и различные технологии, облегчающие и ускоряющие создание программы.

1.4.1 Установка Lazarus в ОС Linux

Для установки Lazarus в окне Synaptic (см. рис. 1.1) необходимо щёлкнуть по кнопке **Найти**. В появившемся окне поиска (см. рис. 1.9) вводим имена необходимых программ (**lazarus, fp, fpc, fpc-source**) и щёлкаем по кнопке **Найти**. Менеджер программ находит программы Lazarus и FreePascal, после чего в окне Synaptic необходимо отметить программы **Lazarus**, **fp**, **fpc**, **fpc-ide** для установки (с помощью контекстного меню или с помощью кнопки **Отметить для обновления**) и начать установку, щёлкнув по кнопке **Применить**. После этого Synaptic предложит установить ещё несколько пакетов, которые необходимы для нормального функционирования Lazarus. С этим предложением нужно согласиться. После этого начнётся процесс загрузки файлов пакетов и установки Lazarus на компьютер. После установки запуск программы осуществляется с помощью команды меню **Разработка** — **Lazarus**[9].

Для установки Lazarus можно не использовать менеджер пакетов Synaptic, а самостоятельно загрузить все необходимые пакеты с сайта проекта[10] и затем вручную их установить. Подробное описание процесса ручной установки можно найти в Интернете[11].

1.4.2 Установка Lazarus под управлением ОС Windows

Рассмотрим особенности установки среды визуального программирования Laza-rus для операционной системы Windows. Перед установкой необходимо скачать установочный файл со страницы загрузки[12]. Установка Lazarus на компьютер осуществляется с помощью Мастера установки. Для того чтобы Мастер

[9] Не исключено, что вызов Lazarus в других дистрибутивах Linux может осуществляться и с помощью другой команды главного меню.

[10] `http://sourceforge.net/project/showfiles.php?group_id=89339`.

[11] `http://freepascal.ru/article//lazarus/20080316091540`.

[12] `http://sourceforge.net/project/showfiles.php?group_id=89339`.

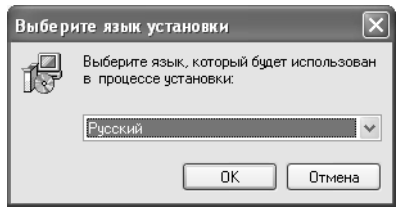

Рис. 1.10. Окно Мастера установки. Выбор языка.

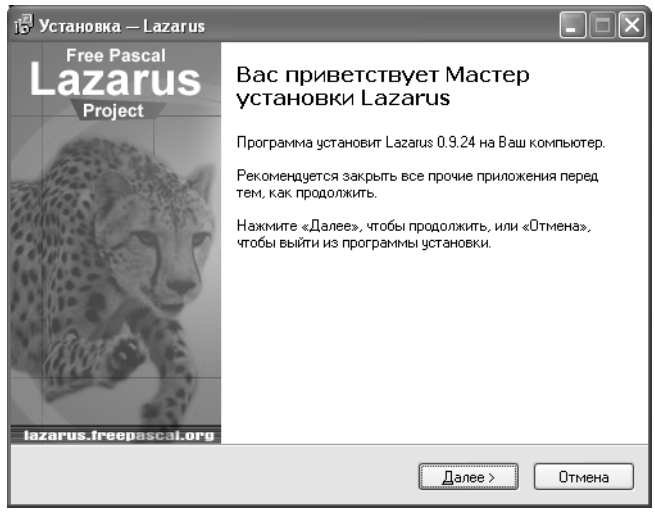

Рис. 1.11. Начало процесса установки Lazarus

начал свою работу, необходимо запустить программу установки Lazarus Setup. Появится диалоговое окно (рис. 1.10), в котором пользователь может выбрать из списка язык для дальнейшего диалога с Мастером установки. Нажатие кнопки **OK** приведёт к появлению следующего окна. Использование кнопки **Отмена** в этом и во всех последующих окнах приведёт к прерыванию работы Мастера установки.

Следующее окно (рис. 1.11) — информационное. Оно сообщает пользователю о начале процесса установки. Нажатие кнопки **Далее** приведёт к следующему шагу Мастера установки.

На следующем этапе (рис. 1.12) необходимо выбрать путь для установки Lazarus. По умолчанию программа будет установлена на диск C. Для выбора иного пути для установки следует воспользоваться кнопкой **Обзор**. В этом и во всех последующих окнах с помощью кнопки **Назад** можно вернуться к предыдущему шагу Мастера установки.

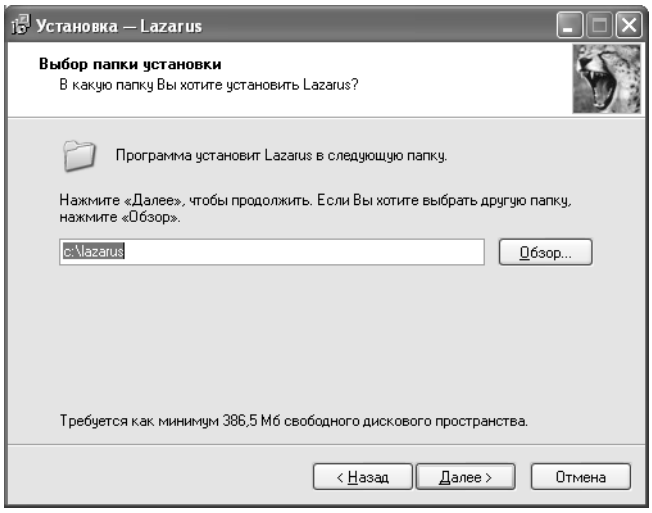

Рис. 1.12. Окно выбора пути для установки Lazarus

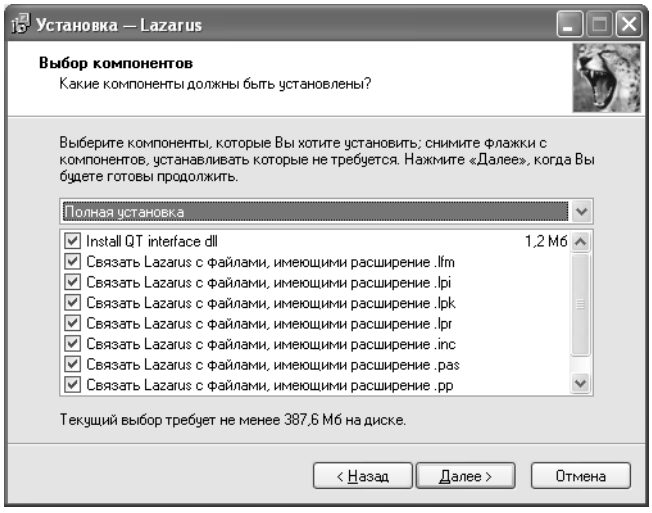

Рис. 1.13. Окно выбора компонентов для установки приложения

Следующий шаг — выбор из полного перечня средств системы необходимых компонентов (рис. 1.13). По умолчанию будут установлены все компоненты системы. Отменить установку компонента можно, если щелчком мыши убрать метку рядом с его именем.

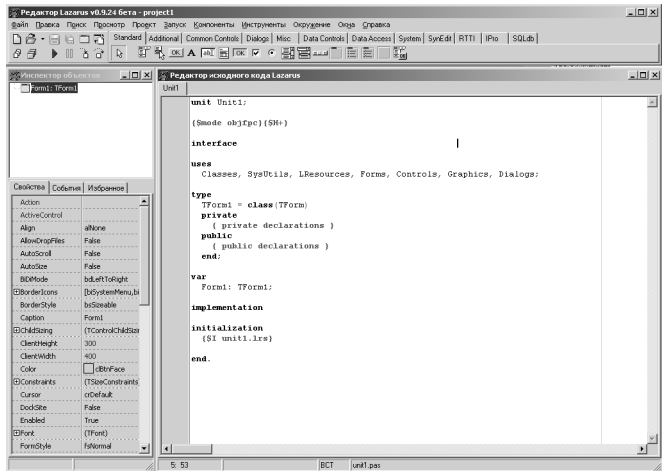

Рис. 1.14. Среда визуального программирования Lazarus

В следующих окнах Мастера установки пользователь может выбрать папку меню Пуск, в которой будет создан ярлык для устанавливаемого приложения[13], и создать на рабочем столе ярлык для приложения Lazarus[14]. Далее Мастер установки сообщит о том, куда будет установлено приложение Lazarus, коков тип установки, какие компоненты системы были выбраны, где будет создан ярлык и будет ли создан значок на рабочем столе. После этого начнётся процесс установки приложения на компьютер. Контролировать установку приложения пользователь может с помощью линейного индикатора, а прервать — кнопкой **Отмена**. Завершается процесс установки щелчком по кнопке **Завершить**. Запуск Lazarus можно осуществить из главного меню командой **Пуск** — **Программы** — **Lazarus** — **Lazarus**.

1.4.3 Среда Lazarus

На рис. 1.14 показано окно, которое появится после запуска Lazarus. В верхней части этого окна размещается *главное меню* и *панель инструментов*. Слева расположено окно *инспектора объектов*, а справа — окно *редактора исходного кода*.

Если свернуть или сдвинуть окно редактора, то станет доступным *окно формы*, представленное на рис. 1.15.

Работу над программой в среде визуального программирования условно можно разбить на две части. Первая — это создание внешнего вида (интерфейса)

[13]По умолчанию ярлык создаётся в меню **Пуск** — **Программы**.

[14]С помощью щелчка мыши установить маркер в поле рядом с командой **Создать значок на рабочем столе**.