Введение

Процесс изучения языка программирования C++ чем-то напоминает приключение первооткрывателя, в частности потому, что этот язык охватывает несколько парадигм программирования, включая объектно-ориентированное программирование (ООП), обобщенное программирование и традиционное процедурное программирование. В пятом издании настоящей книги язык описывался как набор стандартов ISO C++, который неформально назывался C++99 и C++03 или иногда — C++99/03. (Версия 2003 была в основном формальным исправлением стандарта 1999 без добавления каких-либо новых возможностей.) С тех пор язык C++ продолжал развиваться. На момент написания данной книги международный комитет по стандартам C++ одобрил новую версию стандарта. Во время разработки этот стандарт имел неформальное название C++0x, а теперь он будет известен как C++11. Большинство современных компиляторов поддерживают C++99/03 достаточно хорошо, и многие примеры в этой книги соответствуют этому стандарту. Однако в некоторых реализациях уже появились многие возможности, описанные новым стандартом, и эти возможности рассматриваются в настоящем издании книги.

В этой книге обсуждается базовый язык C и текущие функциональные средства C++, что делает ее самодостаточной. В ней представлены основы языка C++, иллюстрируемые с помощью коротких и точных программ, которые легко скопировать для дальнейших экспериментов. Вы узнаете о вводе-выводе, о решении повторяющихся задач и возможностях выбора, о способах обработки данных и о функциях. Будут описаны многие средства C++, которые были добавлены к языку C, включая перечисленные ниже:

- классы и объекты;
- наследование;
- полиморфизм, виртуальные функции и идентификация типов во время выполнения (RTTI);
- перегрузка функций;
- ссылочные переменные;
- обобщенное (или не зависящее от типов) программирование, обеспечиваемое шаблонами и стандартной библиотекой шаблонов (STL);
- механизм исключений для обработки ошибочных условий;
- пространства имен для управления именами функций, классов и переменных.

Принятый подход

Методика изложения материала в этом учебнике обладает рядом преимуществ. Книга написана в духе традиций, которые сложились почти двадцать лет назад с момента выхода книги *Язык программирования C. Лекции и упражнения*, и воплощает в себе следующие принципы.

- Учебник должен быть легко читаемым и удобным в качестве руководства.
- Предполагается, что читатель еще не знаком с принципами программирования, имеющими отношение к теме учебника.
- Изложение материала сопровождается небольшими простыми примерами, которые читатель может выполнить самостоятельно. Примеры способствуют пониманию изложенного материала.

- Концепции в учебнике проясняются с помощью иллюстраций.
- Учебник должен содержать вопросы и упражнения, которые позволят читателю закрепить пройденный материал. Благодаря этому, учебник можно использовать как для самостоятельного, так и для коллективного обучения.

Книга, написанная в соответствии с этими правилами, поможет разобраться во всех тонкостях языка программирования C++ и научит использовать его для решения конкретных задач.

- В учебнике содержится концептуальное руководство по использованию конкретных функциональных средств, таких как применение открытого наследования для моделирования отношений *является*.
- В учебнике приведены примеры распространенных стилей и приемов программирования на языке C++.
- В учебнике имеется большое количество врезок, в том числе советы, предостережения, памятки и примеры из практики.

Автор и редакторы этого учебника приложили максимум усилий, чтобы он получился простым, материал был понятным, и вы остались довольными прочитанным. Мы стремились к тому, чтобы после изучения предложенного материала вы смогли самостоятельно писать надежные и эффективные программы и получать удовольствие от этого занятия.

Примеры кода, используемые в книге

Учебник изобилует примерами кода, большинство из которых являются завершенными программами. Как и в предыдущих изданиях, примеры написаны на обобщенном языке C++, поэтому ни один из них не привязан к определенному типу компьютера, операционной системе или компилятору. Примеры тестировались в системах Windows 7, Macintosh OS X и Linux. В некоторых программах используются средства C++11, и они требуют компиляторов, поддерживающих эти средства, но остальные программы должны работать в любой системе, совместимой с C++99/03.

Код завершенных программ, рассмотренных в этой книге, доступен для загрузки на веб-сайте издательства.

Организация книги

Эта книга содержит 18 глав и 10 приложений, которые кратко описаны ниже.

- **Глава 1, “Начало работы с C++”.** В главе 1 рассказывается о том, как Бьярне Страуструп создал язык программирования C++, добавив к языку C поддержку объектно-ориентированного программирования. Вы узнаете об отличиях между процедурными языками программирования, такими как C, и объектно-ориентированными языками, примером которых является C++. Вы узнаете, как комитетами ANSI/ISO был разработан и утвержден стандарт C++. В главе также рассматривается порядок создания программы на C++ с учетом особенностей множества современных компиляторов C++. В конце главы приведены соглашения, используемые в этой книге.
- **Глава 2, “Приступаем к изучению C++”.** В главе 2 подробно объясняется процесс написания простых программ на C++. Вы узнаете о роли функции `main()` и о некоторых разновидностях операторов, используемых в программах C++. Для операций ввода-вывода в программах вы будете использовать предопределенные объек-

ты cout и cin, а еще вы научитесь создавать и использовать переменные. В конце главы вы познакомитесь с функциями — программными модулями языка C++.

- **Глава 3, "Работа с данными".** Язык программирования C++ предоставляет встроенные типы для хранения двух разновидностей данных — целых чисел (чисел без дробной части) и чисел с плавающей точкой (чисел с дробной частью). Чтобы удовлетворить разнообразные требования программистов, для каждой категории данных в C++ предлагается несколько типов. В главе 3 рассматриваются как сами эти типы, так и создание переменных и написание констант различных типов. Вы также узнаете о том, как C++ обрабатывает явные и неявные преобразования одного типа в другой.
- **Глава 4, "Составные типы".** Язык C++ позволяет строить на основе базовых встроенных типов данных более развитые типы. Самой сложной формой являются классы, о которых пойдет речь в главах 9–13. В главе 4 рассматриваются другие формы, включая массивы, которые хранят множество значений одного и того же типа; структуры, хранящие несколько значений разных типов; и указатели, которые идентифицируют ячейки памяти. Вы узнаете о том, как создавать и хранить текстовые строки и как обрабатывать текстовый ввод-вывод за счет использования символьных массивов с стиле C и класса string из C++. Наконец, будут описаны некоторые способы обработки выделения памяти в C++, в том числе применение операций new и delete для явного управления памятью.
- **Глава 5, "Циклы и выражения отношений".** Программы часто должны выполнять повторяющиеся действия, и для этих целей в C++ предусмотрены три циклических структуры: цикл for, цикл while и цикл do while. Такие циклы должны знать, когда им необходимо завершаться, и операции отношения C++ позволяют создавать соответствующие проверочные выражения. В главе 5 вы узнаете, как создавать циклы, которые читают входные данные и обрабатывают их символ за символом. Наконец, вы научитесь создавать двумерные массивы и применять вложенные циклы для их обработки.
- **Глава 6, "Операторы ветвления и логические операции".** Поведение программы будет интеллектуальным, если она сможет адаптироваться к различным ситуациям. В главе 6 рассказывается об управлении ходом выполнения программы с помощью операторов if, if else и switch, а также условных операций. Вы узнаете, как использовать условные операции для проверки с целью принятия решений. Кроме этого, вы ознакомитесь с библиотекой функций cctype, которая предназначена для оценки символьных отношений, таких как проверка, является ли данный символ цифрой или непечатаемым символом. В конце главы дается краткий обзор файлового ввода-вывода.
- **Глава 7, "Функции как программные модули C++".** Функции являются базовыми строительными блоками в программировании на языке C++. Основное внимание в главе 7 сосредоточено на возможностях, которые функции C++ разделяют с функциями C. В частности, будет представлен общий формат определения функций и показано, как с помощью прототипов функций можно увеличить надежность программ. Вы научитесь создавать функции для обработки массивов, символьных строк и структур. Кроме того, вы узнаете о рекурсии, которая возникает, когда функция обращается к самой себе, а также о том, как с помощью функции можно реализовать стратегию "разделяй и властвуй". Наконец, вы познакомитесь с указателями на функции, благодаря которым одна функция может с помощью аргумента пользоваться другой функцией.

- **Глава 8, "Дополнительные сведения о функциях".** В главе 8 будет рассказано о новых средствах C++, доступных для функций. Вы узнаете, что такое встроенные функции, с помощью которых можно ускорить выполнение программы за счет увеличения ее размера. Вы будете работать со ссылочными переменными, которые предлагают альтернативный способ передачи информации функциям. Аргументы по умолчанию позволяют функции автоматически задавать значения тем аргументам, которые при вызове функции не указаны. Перегрузка функций позволяет создавать функции, которые имеют одно и то же имя, но принимают разные списки аргументов. Все эти средства часто используются при проектировании классов. Вы узнаете также о шаблонах функций, благодаря которым можно строить целые семейства связанных функций.
- **Глава 9, "Модели памяти и пространства имен".** В главе 9 рассматриваются вопросы построения программ, состоящих из множества файлов. В ней будут представлены варианты выделения памяти и методы управления памятью; вы узнаете, что такое область видимости, связывание и пространства имен, которые определяют, каким частям программы будут известны та или иная переменная.
- **Глава 10, "Объекты и классы".** Класс представляет собой определяемый пользователем тип, а объект (такой как переменная) является экземпляром класса. В главе 10 вы узнаете о том, что такое объектно-ориентированное программирование, и как проектируются классы. Объявление класса описывает информацию, хранящуюся в объекте класса, и операции (методы класса), разрешенные для объектов класса. Некоторые части объекта будут видимыми внешнему миру (открытая часть), а другие — скрытыми (закрытая часть). В момент создания и уничтожения объектов инициируются специальные методы класса, называемые конструкторами и деструкторами. В этой главе вы узнаете об этих и других особенностях классов и получите представление об использовании классов для реализации абстрактных типов данных, таких как стек.
- **Глава 11, "Работа с классами".** В главе 11 продолжается знакомство с классами. Прежде всего, будет рассмотрен механизм перегрузки операций, который позволяет определять, каким образом операции вроде + будут работать с объектами класса. Вы узнаете о дружественных функциях, которые могут получать доступ к данным класса, в общем случае недоступных извне. Будет показано, как некоторые конструкторы и функции-члены перегруженных операций могут быть использованы для управления преобразованием одного типа класса в другой.
- **Глава 12, "Классы и динамическое выделение памяти".** Часто требуется, чтобы член класса указывал на динамически выделенную память. Если в конструкторе класса используется операция `new` для динамического выделения памяти, то должен быть предусмотрен соответствующий деструктор, явный конструктор копирования и явная операция присваивания. В главе 12 будет показано, как это можно сделать, и описано поведение функций-членов, которые генерируются неявным образом, если явные описания не предоставлены. Опыт работы с классами расширяется на использование указателей на объекты и моделирование очередей.
- **Глава 13, "Наследование классов".** Одним из самых мощных средств объектно-ориентированного программирования является наследование, благодаря которому производный класс наследует возможности базового класса, позволяя повторно использовать код базового класса. В главе 13 обсуждается открытое наследование, моделирующее отношения *является*, при которых производный

объект рассматривается как специальный случай базового объекта. Например, физик является специальным случаем ученого. Некоторые отношения наследования являются полиморфными, что означает возможность писать код, используя несколько связанных между собой классов, для которых метод с одним и тем же именем может реализовывать поведение, зависящее от типа объекта. Для реализации такого поведения необходимо применять новый вид функции-члена — виртуальную функцию. Иногда для отношений наследования лучше всего использовать абстрактные базовые классы. В этой главе будут рассмотрены все эти вопросы с описанием ситуаций, в которых открытое наследование имеет смысл.

- **Глава 14, "Повторное использование кода в C++".** Открытое наследование — это всего лишь один из способов повторного использования кода. В главе 14 рассматриваются другие варианты. Ситуация, при которой члены одного класса являются объектами другого класса, называется включением. Включение можно использовать для моделирования отношений *содержит*, при которых один класс имеет компоненты, принадлежащие другому классу (например, автомобиль содержит двигатель). Для моделирования таких отношений можно использовать закрытое и защищенное наследование. Вы узнаете о каждом способе и об отличиях разных подходов. Кроме того, будут описаны шаблоны классов, которые позволяют определить класс посредством некоторого неуказанного обобщенного типа, а затем использовать шаблон для создания специфических классов в терминах определенных типов. Например, с помощью шаблона стека можно создать стек целых чисел и стек строк. Наконец, вы узнаете о множественном открытом наследовании, при котором один класс может быть произведен из более чем одного класса.
- **Глава 15, "Друзья, исключения и многое другое".** В главе 15 речь пойдет о так называемых "друзьях", а именно — о дружественных классах и дружественных функциях-членах. Здесь вы найдете сведения о некоторых нововведениях C++, начиная с исключений, которые предлагают механизм обработки необычных ситуаций, возникающих при выполнении программы (например, неподходящие значения аргументов функции или нехватка памяти). В этой главе вы узнаете, что собой представляет механизм RTTI (идентификация типов во время выполнения). В заключительной части главы будет рассказано о безопасных альтернативах неограниченному приведению типов.
- **Глава 16, "Класс `string` и стандартная библиотека шаблонов".** В главе 16 рассказывается о некоторых полезных библиотеках классов, недавно добавленных к языку. Класс `string` является удобной и мощной альтернативой традиционным строкам в стиле C. Класс `auto_ptr` помогает управлять динамически выделяемой памятью. Библиотека STL содержит множество обобщенных контейнеров, включая шаблонные представления массивов, очередей, списков, множеств и карт. Она также предоставляет эффективную библиотеку обобщенных алгоритмов, которые можно использовать с контейнерами STL и обычными массивами. Шаблонный класс `valarray` обеспечивает поддержку для числовых массивов.
- **Глава 17, "Ввод, вывод и файлы".** В главе 17 рассматривается ввод-вывод в C++ и обсуждаются вопросы форматирования вывода. Вы узнаете, как использовать методы классов для определения состояния потоков ввода и вывода, чтобы проверить, например, присутствует во входных данных несоответствие типов либо условие достижения конца файла. Чтобы произвести классы для управле-

ния файловым вводом и выводом, в C++ используется наследование. В главе будет показано, как открывать файлы для ввода и вывода, как добавлять данные в файл, как работать с бинарными файлами и как получить произвольный доступ к файлу. Напоследок вы узнаете о том, как применять стандартные методы ввода-вывода для чтения и записи строк.

- **Глава 18, "Новый стандарт C++".** Глава 18 начинается с обзора множества функциональных средств C++11, которые были введены в предшествующих главах, в том числе новые типы, унифицированный синтаксис инициализации, автоматическое выведение типов, новые интеллектуальные указатели и перечисления с областью видимости. Затем в главе обсуждается новый тип ссылки rvalue и показано, как он используется для реализации нового средства под названием *семантика переноса*. Кроме того, рассматриваются новые возможности классов, лямбда-выражения и шаблоны с переменным числом аргументов. Наконец, в главе даны описания многих новых средств, которые не упоминались в предшествующих главах книги.
- **Приложение А, "Основания систем счисления".** В приложении А обсуждаются восьмеричные, шестнадцатеричные и двоичные числа.
- **Приложение Б, "Зарезервированные слова C++".** В приложении Б приведен список ключевых слов C++.
- **Приложение В, "Набор символов ASCII".** В приложении В приведен набор символов ASCII вместе с его десятичным, восьмеричным, шестнадцатеричным и двоичным представлениями.
- **Приложение Г, "Приоритеты операций".** В приложении Г перечислены операции C++ в порядке убывания приоритетов.
- **Приложение Д, "Другие операции".** В приложении Д приводятся сведения об операциях C++, которые не были рассмотрены в основном тексте (например, поразрядные операции).
- **Приложение Е, "Шаблонный класс `string`".** В приложении Е приводятся сведения о методах и функциях класса `string`.
- **Приложение Ж, "Методы и функции стандартной библиотеки шаблонов".** В приложении Ж приводятся сведения о методах контейнеров STL и функциях общих алгоритмов STL.
- **Приложение З, "Рекомендуемая литература и ресурсы в Интернете".** В приложении З предложен список книг и ресурсов, которые можно использовать для дальнейшего изучения C++.
- **Приложение И, "Переход к стандарту ANSI/ISO C++".** В приложении И представлено руководство по переносу кода на C и старых реализаций C++ в код на ANSI/ISO C++.
- **Приложение К, "Ответы на вопросы для самоконтроля".** В приложении К содержатся ответы на вопросы для самоконтроля, приведенные в конце каждой главы.

Примечание для преподавателей

Одна из целей этого издания заключалась в том, чтобы предложить книгу, которую можно было бы использовать либо в качестве самоучителя, либо в качестве учебника.

Ниже перечислены некоторые характеристики, которые позволяют рассматривать эту книгу как учебник.

- Книга описывает обобщенный язык программирования C++, т.е. не зависит от конкретной реализации.
- Предложенный материал соответствует стандарту ANSI/ISO C++ и включает обсуждение шаблонов, библиотеки STL, класса `string`, исключений, RTTI и пространств имен.
- Для чтения книги не требуются предварительные знания языка C (хотя определенные навыки в программировании желательны).
- Темы упорядочены таким образом, что первые главы могут быть изучены довольно быстро как обзорные главы на курсах, слушатели которых знакомы с языком C.
- В конце каждой главы предлагаются вопросы для самоконтроля и упражнения по программированию. В приложении К даны ответы на вопросы для самоконтроля.
- В книге присутствуют темы, подходящие для курсов по вычислительной технике, включая абстрактные типы данных, стеки, очереди, простые списки, эмуляции, обобщенное программирование и применение рекурсии для реализации стратегии "разделяй и властвуй".
- Большинство глав имеют такой объем, чтобы их можно было изучить в течение одной недели или даже быстрее.
- В книге обсуждается, *когда* можно использовать определенные функциональные средства, и *каким образом* их использовать. Например, открытое наследование связывается с отношениями *содержит*, композиция и закрытое наследование — с отношениями *является*, кроме того, рассматриваются вопросы о том, когда использовать виртуальные функции, а когда — нет.

Соглашения, используемые в этой книге

Для выделения различных частей текста в книге используются следующие типографские соглашения.

- Строки кода, команды, операторы, переменные, имена файлов и вывод программ выделяется моноширинным шрифтом:

```
#include <iostream>
int main()
{
   using namespace std;
   cout << "What's up, Doc!\n";
   return 0;
}
```

- Входные данные для программ, которые вы должны вводить самостоятельно, представлены **моноширинным шрифтом с полужирным начертанием**:

```
Please enter your name:
Plato
```

- Заполнители в описаниях синтаксиса выделены *моноширинным шрифтом с курсивным начертанием*. Каждый такой заполнитель должен заменяться реальным именем файла, параметром или любым другим элементом, который он представляет.
- Для выделения новых терминов используется *курсив*.

Врезка

Врезка содержит более детальное обсуждение или дополнительные сведения по текущему вопросу.

Совет

Здесь вы найдете краткие полезные советы и рекомендации по программированию.

Внимание!

Здесь вы встретите предупреждения о потенциальных ловушках.

На заметку!

Здесь предоставляются разнообразные комментарии, которые не подпадают ни под одну из указанных выше категорий.

Системы, на которых разрабатывались примеры для книги

Примеры на C++11 в этой книге разрабатывались с использованием Microsoft Visual C++ 2010 и Cygwin с Gnu g++ 4.5.0 в среде 64-разрядной ОС Windows 7. Остальные примеры тестировались в этих системах, а также в системе iMac с применением g++ 4.2.1 под OS X 10.6.8 и в системе Ubuntu Linux с использованием g++ 4.4.1. Большинство примеров, предшествующих C++11, изначально разрабатывались с помощью Microsoft Visual C++ 2003 и Metrowerks CodeWarrior Development Studio 9 в среде Windows XP Professional и тестировались с применением компиляторов командной строки Borland C++ 5.5 и GNU gpp 3.3.3 в той же системе, компиляторов Comeau 4.3.3 и GNU g++ 3.3.1 в системе SuSE 9.0 Linux и Metrowerks Development Studio 9 на Macintosh G4 под OS 10.3.

Для любого программиста язык C++ открывает целый мир возможностей; изучайте его и наслаждайтесь работой!