ОГЛАВЛЕНИЕ

Вступление
О КНИГЕ И ЯЗЫКЕ C++

Замечательная идея! Что ж она мне самому в голову не пришла?

из к/ф «Ирония судьбы, или С легким паром»

Вниманию читателя предлагается книга о языке программирования C++. Пособие для тех, кто не знаком (или мало знаком) с языком C++ и хочет быстро научиться на нем программировать.

Собственно о книге

Издание состоит из тематически подобранных примеров и задач, которые перекрывают все основные разделы и вопросы, важные с точки зрения изучения основ языка C++. Здесь хочется отметить два важных обстоятельства.

Во-первых, опыт показывает, что наиболее успешно материал усваивается, если он иллюстрируется примерами. Более того, иногда очень сложно объяснить определенную концепцию или подход, если они не подкреплены практическим материалом. В этом смысле для книги выбран оптимальный способ представления информации.

Во-вторых, предполагается, что при работе с материалом книги читатель проявит некоторую настойчивость и готовность к углубленному анализу программных кодов. С другой стороны, автор приложил максимальные усилия для того, чтобы сделать содержимое книги простым и наглядным даже для неподготовленного читателя.

Последняя глава в определенном смысле особенная. В ней собраны некоторые математические задачи вычислительного характера. Они ориентированы на большую самостоятельность читателя в плане анализа программных кодов (по сравнению с примерами из других глав книги). Данная глава включена в книгу, поскольку представленные в ней задачи

рассматриваются в рамках университетского лекционного курса по вычислительной математике.

Для лучшего закрепления материала в конце первых четырех глав приводится список задач для самостоятельного решения. Они концептуально близки к тем задачам и примерам, что рассматривались в основной части соответствующей главы. Некоторые задачи очень просты, но встречаются и такие, для решения которых необходимо проявить смекалку. Также стоит отметить, что для части задач, предлагаемых для самостоятельного решения, в книге (обычно в следующих главах после той, где предлагается задача) приводятся решения. Возможно, контекст задачи при решении немного изменяется, но узнать знакомые «ноты» все же можно. Сделано это специально, поскольку, как показывает опыт, решить одну и ту же задачу разными методами бывает намного полезнее, чем просто решать разные задачи.

В пятой и шестой главах, где обсуждаются принципы объектно-ориентированного программирования, список задач для самостоятельного решения также есть, но он намного скромнее. Предлагаемые там задачи носят «эвристический» характер и сформулированы в общем виде. Это намеренная позиция, поскольку в данном случае важен контекст постановки задачи. Другими словами, частью задачи, которую предстоит решить читателю, является, в том числе, и ее окончательная формулировка.

В главах с пятой по восьмую выделяются некоторые «идеологические» направления на тот случай, если читатель решит самостоятельно изучать соответствующие аспекты программирования (рекомендации для самостоятельной работы). Задачи для самостоятельного решения в седьмой и восьмой главах не предлагаются, поскольку материал там сложный и некоторые темы представлены скорее в режиме ознакомления.

Наконец, в последней главе, где рассматриваются математические вычислительные задачи, нет ни заданий для самостоятельного решения, ни рекомендаций для самостоятельной работы. Причина в том, что глава основательно ориентирована на самостоятельное изучение нетривиального материала, поэтому перегружать ее нежелательно.

Язык программирования C++

C++ — один из наиболее популярных языков программирования. Невозможно представить себе профессионального программиста, который

не знал бы его. В этом смысле выбор языка C++ для изучения представляется весьма удачным.

Помимо непосредственной практической пользы от умения составлять программы на этом языке, есть еще и немаловажный методологический аспект. Связан он с исключительной гибкостью и богатством C++. После его изучения намного легче «брать на вооружение» прочие языки программирования. Но как бы там ни было, C++ на сегодня востребован, и в ближайшее время это вряд ли изменится.

Среда разработки

Нет недостатка в программном обеспечении для работы с программными кодами на C++. На данный момент существует много качественных средств разработки — как коммерческих, так и бесплатных. Обычно используют *среды разработки*, в состав которых входит редактор кодов, отладчик, обычно компилятор и ряд других вспомогательных утилит. Читатель, безусловно, волен выбрать ту среду разработки, которая ему наиболее симпатична. Рассматриваемые в основной части книги коды универсальны и не привязаны к какой-то конкретной среде. Вместе с тем важно отметить, что тестировались программы в Visual Studio Express 2013. Желающие могут загрузить (бесплатно, но с регистрацией) данную среду разработки с сайта компании Microsoft.

Об авторе

Автор книги — *Васильев Алексей Николаевич*, доктор физико-математических наук, профессор кафедры теоретической физики физического факультета Киевского национального университета имени Тараса Шевченко. Автор книг по программированию и математическому моделированию. Сфера научных интересов: физика жидкостей, биофизика, синергетика, математические методы в экономике, моделирование социально-политических процессов, математическая лингвистика.

Обратная связь

Вы можете высказать свои замечания и предложения по поводу книги по электронной почте `alex@vasilev.kiev.ua`.

Файлы для скачивания

Дополнительные материалы можно скачать по адресу https://eksmo.ru/C++_codes.zip

Благодарности

Автор выражает искреннюю признательность своим читателям за интерес к книгам. Понимание того, что книги востребованы читателями, является наилучшим стимулом для их написания.

Глава 1
ПРОСТЫЕ ПРОГРАММЫ

Софистика, пастор, софистика!
из к/ф «Семнадцать мгновений весны»

В этой главе рассматриваются наиболее простые программы. Кроме прочего, мы узнаем:

- как организована программа в C++;
- как осуществляется ввод и вывод данных через консольное окно;
- как объявляются переменные;
- как выполняются простые арифметические операции,

ну и кое-что еще.

Первая программа

Далее рассматривается небольшая программа, которая отображает приветствие в консольном окне. Соответствующий программный код приведен в листинге 1.1.

Листинг 1.1. Первая программа

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main(){
   // Изменение кодировки консоли:
   system("chcp 1251>nul");
   // Отображение сообщения в консоли:
   cout<<"Программируем на C++!"<<endl;
   // Задержка консольного окна:
   system("pause>nul");
   return 0;
}
```

При выполнении программы получаем такой результат:

Результат выполнения программы (из листинга 1.1)

```
Программируем на C++!
```

Соответствующее сообщение появляется в консольном окне.

Рассмотрим и проанализируем программный код. Прежде всего, выделим основные блоки в программе. Фактически он там один: это тело *главной функции* программы. Она называется `main()` и описана в соответствии с таким шаблоном:

```
int main(){
   // Код главной функции программы
}
```

Собственно тело функции выделяется с помощью пары фигурных скобок: открывающей `{` и закрывающей `}`. То, что между этими скобками — код функции `main()`. Ключевое слово `int` в описании функции означает, что функция возвращает результат, и результат функции является целым числом.

Выполнение программы в C++ — это выполнение главной функции `main()`. Проще говоря, чтобы понять, какие команды выполняются при выполнении программы, следует обратиться к функции `main()`. В данном случае в теле функции несколько (точнее, семь) строчек кода, причем три из них — *комментарии*.

Комментарии начинаются с двойной косой черты и при компиляции программы игнорируются. Назначение комментария — объяснить тому, кто анализирует программный код, назначение тех или иных команд. Комментарии предназначены для человека, а не для компьютера. Необходимости в использовании комментариев в программном коде нет, но их наличие значительно облегчает задачу программиста.

ТЕОРИЯ

В C++ используются однострочные и многострочные комментарии. Однострочный комментарий начинается с двойной косой черты `//`. Все, что находится справа — комментарий. Однострочный комментарий размещается в одной строке.

Многострочные комментарии, как несложно догадаться, состоят из нескольких строк. Начало выделяется комбинацией символов `/*`,

а конец — */. Все, что находится между инструкциями /* и */ является комментарием.

Две команды в теле функции main() имеют технический, «вспомогательный» характер. Их наличие в программном коде связано с особенностями функционирования консольного окна и необходимостью отображать в нем кириллический текст. Речь о командах system("chcp 1251>nul") и system("pause>nul"). Первая переводит консоль в режим использования кодировки *Windows-1251*, а вторая нужна для того, чтобы после выполнения программы консольное окно не закрывалось сразу, а задерживалось на экране до нажатия пользователем клавиши **Enter** (или какой-то другой).

ПОДРОБНОСТИ

При работе с операционной системой Windows и вводе/выводе информации через консольное окно пользователь, скорее всего, столкнется с двумя проблемами. Первая возникает при отображении кириллического текста в консольном окне.

Фундаментальная сложность связана с тем, что редактор кодов, в котором набирается код программы, имеет кодировку, отличную от кодировки консольного кона. Поэтому кириллический текст, набранный в редакторе кодов, будет нормально выглядеть в окне редактора кодов, но совершенно неприлично при отображении его в консоли. Та же проблема с вводом текста: если в консоли вводится кириллический текст, то программой он будет считан как набор странных символов.

Проблему можно решить, установив в окне консоли такую же кодировку, как в окне редактора кодов. Сделать это можно вручную или непосредственно через программный код. Для этого в начале программного кода помещается команда system("chcp 1251>nul").

Еще одна проблема связана с тем, что после завершения выполнения программы консольное окно, куда осуществляется вывод, автоматически закрывается. Происходит все очень быстро — настолько, что пользователь не успевает заметить, какая информация выводилась в консольное окно при выполнении программы. Чтобы консоль не закрывалась по завершении выполнения программы, добавляем в программный код команду system("pause>nul").

Команды system("chcp 1251>nul") и system("pause>nul") представляют собой вызов функции system(). Чтобы функция system() была доступна, подключаем заголовок <cstdlib>.

Аргументом функции system() передается текстовая строка, содержащая команду, которая должна быть выполнена в консоли.

В частности, команда `pause` для консоли означает задержку консольного окна, а команда `chcp 1251` используется для применения соответствующей кодировки. В принципе, в программном коде можно использовать команды `system("chcp 1251")` и `system("pause")`, однако в этом случае в консольное окно выводятся сообщения соответственно об установке кодировки и о необходимости нажать кнопку для закрытия консольного кона. Чтобы этого не было, используем инструкцию `nul`. Она указывается через символ `>` после собственно команды, которую необходимо выполнить в консоли.

Пожалуй, самая важная команда в теле функции `main()` — это `cout<<"Программируем на C++!"<<endl`. Именно она решает основную задачу по отображению сообщения в консольном окне, ради чего, собственно, и создавалась программа. Основу команды составляет *оператор вывода* `<<`. Оператором вывода данные, указанные справа, отображаются в устройстве, ссылка на которое указана слева от оператора. Например, команда `cout<<"Программируем на C++!"` означает, что необходимо текст `"Программируем на C++!"` вывести в устройство, «спрятанное» за идентификатором `cout`.

По умолчанию идентификатор `cout` означает консоль (*cout* от *console output*). Текст, как несложно догадаться, заключается в двойные кавычки. Операторов вывода в одной команде разрешается использовать несколько. Инструкция `endl` (от *end of line*) используется для перевода курсора в новую строку в окне вывода (консоли). Поэтому вся команда `cout<<"Программируем на C++!"<<endl` означает следующее: в консольное окно необходимо вывести текст `"Программируем на C++!"`, после чего выполняется переход к новой строке.

НА ЗАМЕТКУ

В принципе, нет крайней необходимости в использовании инструкции `endl` для перехода к новой строке. Вместо команды `cout<<"Программируем на C++!"<<endl` вполне сгодилась бы и команда `cout<<"Программируем на C++!"`.

Последняя в теле функции `main()` команда `return 0` означает завершение выполнения функции и возвращение результатом функции значения `0`. Значение `0` дает знать операционной системе, что работа функции завершилась в нормальном режиме, без ошибки. Последней командой в функции `main()` обычно является команда `return 0`.

НА ЗАМЕТКУ

Также хочется отметить, что команды в C++ заканчиваются точкой с запятой.

Мы разобрали код главной функции, но есть еще несколько инструкций в самом начале программы, до начала описания главной функции.

Инструкции `#include <iostream>` и `#include <cstdlib>` представляют собой директивы препроцессора и используются для включения в программный код заголовочных файлов, которые содержат важную информацию, необходимую для выполнения программы.

Фактически, речь идет об использовании базовых стандартных библиотек, необходимых для выполнения программы (`<iostream>` соответствует стандартной библиотеке ввода/вывода, а `<cstdlib>` соответствует стандартной общей библиотеке).

ПОДРОБНОСТИ

Стандартная библиотека ввода/вывода содержит определение таких идентификаторов, как `cout` (консольное устройство) и `cin` (клавиатура — устройство ввода). Стандартная общая библиотека содержит определение, кроме прочего, функции `system()`.

Инструкция `using namespace std` представляет собой команду для использования стандартного *пространства имен* `std`.

ТЕОРИЯ

Пространство имен — это некое абстрактное хранилище, позволяющее избегать конфликта имен. При составлении программного кода нам необходимо указать, какое именно пространство имен будет использовано для «хранения» названий создаваемых в программе утилит. Мы будем использовать стандартное пространство имен, которое называется `std`.

Собственно на этом мы заканчиваем анализ программного кода. В завершение примера отметим, что при написании программ мы в дальнейшем планируем использовать следующий шаблонный код (без учета комментариев):

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main(){
   system("chcp 1251>nul");
   // Код главной функции
   system("pause>nul");
   return 0;
}
```

Если же читатель решит проблему с кодировкой и задержкой консольного окна способом, отличным от представленного выше, то шаблонный код еще проще:

```
#include <iostream>
using namespace std;
int main(){
   // Код главной функции
   return 0;
}
```

Теперь переходим к рассмотрению следующего примера. В нем мы познакомимся с *оператором ввода* (узнаем, как вводится информация с клавиатуры) и *переменными*.

Знакомство с переменными

Далее мы рассмотрим задачу о переводе расстояния, указанного в милях, в значение, указанное в километрах. При ее решении нам предстоит вводить данные с клавиатуры и передавать их в программу для выполнения необходимых вычислений.

> **НА ЗАМЕТКУ**
>
> Британская статутная миля — мера длины, равная 1609344 миллиметрам или 1,609344 километра. Чтобы перевести значение, указанное в милях, в километры, достаточно умножить значение в милях на 1,609344 .

Программа достаточно простая: сначала выводится запрос на ввод пользователем расстояния в милях, после чего введенное значение считывается, выполняются необходимые вычисления, а результат вычислений отображается в консольном окне.

В процессе вычислений нам понадобится куда-то записывать вводимое пользователем значение, равно как и результаты вычислений. Для этого в программе используются *переменные*.

ТЕОРИЯ

Переменная — именованная область памяти, в которую можно записать значение и из которой можно прочитать значение. В C++ переменные перед использованием объявляются. Для каждой переменной указывается тип значения, которое может храниться в переменной. Тип переменной определяется через ключевое слово — идентификатор типа. Наиболее актуальные базовые типы: `int` (целые числа), `double` (действительные числа), `char` (символы).

В программе переменная объявляется в любом месте (но до первого ее использования). Область доступности переменной определяется блоком, в котором она объявлена. Блок, в свою очередь, ограничивается парой фигурных скобок. Если переменная объявлена в главной функции программы, она доступна в любом месте главной функции.

Помимо переменных, нередко используются **константы**. От переменных они отличаются тем, что значение константы изменить нельзя. Константы описываются так же, как переменные, но с идентификатором `const`.

Теперь рассмотрим программный код, представленный в листинге 1.2.

Листинг 1.2. Перевод миль в километры

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main(){
   // Изменение кодировки консоли:
   system("chcp 1251>nul");
   // Константа определяет, сколько в одной
   // миле километров:
   const double kmInMile=1.609344;
```