

Содержание

Об авторе	16
Введение	17
Что нового в третьем издании	17
О версии C++	18
Как работать с этой книгой	18
Если вы работаете под управлением Windows	18
Программный код — из Web-пространства	19
Что еще почитать	19
Ждем ваших отзывов!	20
Глава 1. Из истории создания C++	21
Истоки C++	22
Создание языка C	22
Предпосылки возникновения языка C++	24
Рождение C++	25
Эволюция C++	26
Что такое объектно-ориентированное программирование	27
Инкапсуляция	27
Полиморфизм	28
Наследование	29
C++ и реализация ООП	29
Связь C++ с языками Java и C#	29
Глава 2. Обзор элементов языка C++	31
Первая C++-программа	32
Ввод текста программы	33
Компилирование программы	33
Выполнение программы	34
Построчный “разбор полетов”	34
Обработка синтаксических ошибок	37
Вторая C++-программа	38
Более реальный пример	39
Новый тип данных	40
Повторим пройденное	41
Функции	41
Программа с двумя функциями	42
Аргументы функций	43
Функции, возвращающие значения	45
Функция <code>main()</code>	46
Общий формат C++-функций	47
Некоторые возможности вывода данных	47
Две простые инструкции	48
Инструкция <code>if</code>	48
Цикл <code>for</code>	49
Блоки кода	51
Точки с запятой и расположение инструкций	52
Практика отступов	52

Ключевые слова C++	52
Идентификаторы в C++	53
Стандартная библиотека C++	54
Глава 3. Основные типы данных	55
Объявление переменных	57
Локальные переменные	57
Формальные параметры	58
Глобальные переменные	59
Модификаторы типов	60
Литералы	63
Шестнадцатеричные и восьмеричные литералы	64
Строковые литералы	65
Управляющие символьные последовательности	65
Инициализация переменных	66
Операторы	68
Арифметические операторы	68
Инкремент и декремент	69
История происхождения имени C++	70
Операторы отношений и логические операторы	71
Выражения	73
Преобразование типов в выражениях	74
Преобразования, связанные с типом <code>bool</code>	74
Приведение типов	75
Использование пробелов и круглых скобок	76
Глава 4. Инструкции управления	77
Инструкция <code>if</code>	78
Условное выражение	79
Вложенные <code>if</code> -инструкции	80
Конструкция <code>if-else-if</code>	81
Цикл <code>for</code>	82
Вариации на тему цикла <code>for</code>	84
Отсутствие элементов в определении цикла	86
Бесконечный цикл	87
Циклы временной задержки	87
Инструкция <code>switch</code>	87
Вложенные инструкции <code>switch</code>	91
Цикл <code>while</code>	91
Цикл <code>do-while</code>	93
Использование инструкции <code>continue</code>	94
Использование инструкции <code>break</code> для выхода из цикла	95
Вложенные циклы	97
Инструкция <code>goto</code>	97
Итак, подведем итоги...	99
Глава 5. Массивы и строки	101
Одномерные массивы	102
На границах массивов погранзаставы нет	104
Сортировка массива	105
Строки	106
Считывание строк с клавиатуры	107

Некоторые библиотечные функции обработки строк	108
Функция <code>strcpy()</code>	109
Функция <code>strcat()</code>	109
Функция <code>strcmp()</code>	110
Функция <code>strlen()</code>	111
Использование признака завершения строки	113
Двумерные массивы	114
Многомерные массивы	115
Инициализация массивов	115
“Безразмерная” инициализация массивов	118
Массивы строк	119
Пример использования массивов строк	120
Глава 6. Указатели	123
Что представляют собой указатели	124
Операторы, используемые с указателями	125
О важности базового типа указателя	126
Присваивание значений с помощью указателей	128
Использование указателей в выражениях	129
Арифметические операции над указателями	129
Сравнение указателей	131
Указатели и массивы	131
Индексирование указателя	134
О взаимозаменяемости указателей и массивов	135
Указатели и строковые литералы	135
Все познается в сравнении	136
Массивы указателей	137
Соглашение о нулевых указателях	140
Указатели и 16-разрядные среды	140
Многоуровневая непрягая адресация	141
Проблемы, связанные с использованием указателей	142
Неинициализированные указатели	143
Некорректное сравнение указателей	143
Не забывайте об установке указателей	144
Глава 7. Функции, часть первая: основы	147
Правила действия областей видимости функций	148
Локальные переменные	148
Объявление переменных в итерационных инструкциях и инструкциях выбора	153
Формальные параметры	154
Глобальные переменные	154
Передача указателей и массивов в качестве аргументов	156
Вызов функций с указателями	156
Вызов функций с массивами	158
Передача функциям строк	160
Аргументы функции <code>main(): argc</code> и <code>argv</code>	162
Передача числовых аргументов командной строки	164
Преобразование числовых строк в числа	165
Инструкция <code>return</code>	166
Завершение функции	166
Возврат значений	167
Функции, которые не возвращают значений (<code>void</code> -функции)	169
Функции, которые возвращают указатели	170

Прототипы функций	171
Подробнее о заголовках	172
Сравнение старого и нового стилей объявления параметров функций	173
Рекурсия	173
Глава 8. Функции, часть вторая: ссылки, перегрузка и использование аргументов по умолчанию	177
Два способа передачи аргументов	178
Как в C++ реализована передача аргументов	178
Использование указателя для обеспечения вызова по ссылке	179
Ссылочные параметры	181
Объявление ссылочных параметров	183
Возврат ссылок	184
Создание ограниченного массива	187
Независимые ссылки	188
Ограничения при использовании ссылок	189
Перегрузка функций	190
Анахронизм в виде ключевого слова <code>overload</code>	193
Аргументы, передаваемые функции по умолчанию	193
Сравнение возможности передачи аргументов по умолчанию с перегрузкой функций	195
Об использовании аргументов, передаваемых по умолчанию	197
Перегрузка функций и неоднозначность	197
Глава 9. Еще о типах данных и операторах	201
Спецификаторы типа <code>const</code> и <code>volatile</code>	202
Спецификатор типа <code>const</code>	202
Спецификатор типа <code>volatile</code>	204
Спецификаторы классов памяти	206
Спецификатор класса памяти <code>auto</code>	206
Спецификатор класса памяти <code>extern</code>	206
Статические переменные	208
Локальные <code>static</code> -переменные	208
Глобальные <code>static</code> -переменные	210
Регистровые переменные	211
Происхождение модификатора <code>register</code>	213
Перечисления	214
Ключевое слово <code>typedef</code>	217
Еще об операторах	218
Поразрядные операторы	218
Поразрядные операторы И, ИЛИ, исключающее ИЛИ и НЕ	219
Операторы сдвига	222
Оператор “знак вопроса”	224
Составные операторы присваивания	225
Оператор “запятая”	226
Несколько присваиваний “в одном”	227
Использование ключевого слова <code>sizeof</code>	227
Динамическое распределение памяти с использованием операторов <code>new</code> и <code>delete</code>	228
Инициализация динамически выделенной памяти	231
Выделение памяти для массивов	232
Динамическое распределение памяти в языке C: функции <code>malloc()</code> и <code>free()</code>	233
Сводная таблица приоритетов C++-операторов	234

Глава 10. Структуры и объединения	237
Структуры	238
Доступ к членам структуры	240
Массивы структур	241
Простой пример инвентаризации склада	241
Передача структур функциям	247
Присваивание структур	248
Использование указателей на структуры и оператора “стрелка”	250
Пример использования указателей на структуры	251
Ссылки на структуры	253
Использование в качестве членов структур массивов и структур	254
Сравнение C- и C++-структур	255
Битовые поля структур	256
Объединения	258
Анонимные объединения	262
Использование оператора <code>sizeof</code> для гарантии переносимости программного кода	263
Переходим к объектно-ориентированному программированию	264
Глава 11. Введение в классы	265
Основы понятия класса	266
Общий формат объявления класса	270
Доступ к членам класса	270
Конструкторы и деструкторы	272
Параметризованные конструкторы	275
Альтернативный вариант инициализации объекта	279
Классы и структуры — родственные типы	280
Сравнение структур с классами	282
Объединения и классы — родственные типы	282
Встраиваемые функции	283
Использование встраиваемых функций в определении класса	285
Массивы объектов	286
Инициализация массивов объектов	288
Указатели на объекты	289
Ссылки на объекты	291
Глава 12. О классах подробнее	293
Функции-“друзья”	294
Перегрузка конструкторов	298
Динамическая инициализация	300
Применение динамической инициализации к конструкторам	300
Присваивание объектов	302
Передача объектов функциям	303
Конструкторы, деструкторы и передача объектов	304
Потенциальные проблемы при передаче параметров	305
Возвращение объектов функциями	308
Потенциальная проблема при возвращении объектов функциями	309
Создание и использование конструктора копии	311
Конструкторы копии и параметры функции	312
Использование конструкторов копии при инициализации объектов	314
Использование конструктора копии при возвращении функцией объекта	315
Конструкторы копии — а нельзя ли найти что-то попроще?	316
Ключевое слово <code>this</code>	317

Глава 13. Перегрузка операторов	319
Перегрузка операторов с использованием функций-членов	320
Использование функций-членов для перегрузки унарных операторов	323
Советы по реализации перегрузки операторов	328
О значении порядка операндов	329
Перегрузка операторов с использованием функций-не членов класса	329
Использование функций-“друзей” для перегрузки унарных операторов	332
Перегрузка операторов отношения и логических операторов	336
Подробнее об операторе присваивания	336
Перегрузка оператора индексации массивов ([])	340
Перегрузка оператора “ () ”	343
Перегрузка других операторов	345
Еще один пример перегрузки операторов	345
Глава 14. Наследование	351
Понятие о наследовании	352
Управление доступом к членам базового класса	355
Использование защищенных членов	357
Использование спецификатора <code>protected</code> для наследования базового класса	361
Об использовании спецификаторов <code>public</code> , <code>protected</code> и <code>private</code>	362
Наследование нескольких базовых классов	362
Конструкторы, деструкторы и наследование	363
Когда выполняются конструкторы и деструкторы	364
Передача параметров конструкторам базового класса	367
Предоставление доступа	370
Чтение C++-графов наследования	372
Виртуальные базовые классы	373
Глава 15. Виртуальные функции и полиморфизм	377
Указатели на производные типы	378
Ссылки на производные типы	381
Виртуальные функции	381
Наследование виртуальных функций	384
Зачем нужны виртуальные функции	386
Простое приложение виртуальных функций	387
Чисто виртуальные функции и абстрактные классы	390
Сравнение раннего связывания с поздним	393
Полиморфизм и пуризм	394
Глава 16. Шаблоны	395
Обобщенные функции	396
Функция с двумя обобщенными типами	399
Явно заданная перегрузка обобщенной функции	399
Перегрузка шаблона функции	401
Использование стандартных параметров в шаблонных функциях	402
Ограничения при использовании обобщенных функций	403
Создание обобщенной функции <code>abs ()</code>	403
Обобщенные классы	404
Пример класса с двумя обобщенными типами данных	407
Создание обобщенного класса безопасного массива	408
Использование в обобщенных классах аргументов, не являющихся типами	409
Использование в шаблонных классах аргументов по умолчанию	411
Явно задаваемые специализации классов	413

Глава 17. Обработка исключительных ситуаций	415
Основы обработки исключительных ситуаций	416
Функции <code>exit()</code> и <code>abort()</code>	419
Перехват исключений классowego типа	421
Использование нескольких <code>catch</code> -инструкций	423
Перехват исключений базового класса	424
Варианты обработки исключений	424
Перехват всех исключений	425
Ограничения, налагаемые на тип исключений, генерируемых функциями	427
Повторное генерирование исключения	428
Обработка исключений, сгенерированных оператором <code>new</code>	430
Альтернативная форма оператора <code>new</code> — <code>nothrow</code>	431
Перегрузка операторов <code>new</code> и <code>delete</code>	432
Перегрузка <code>nothrow</code> -версии оператора <code>new</code>	436
Глава 18. C++-система ввода-вывода	437
Сравнение старой и новой C++-систем ввода-вывода	438
Потоки C++	438
Встроенные C++-потоки	440
Классы потоков	440
Перегрузка операторов ввода-вывода	441
Создание перегруженных операторов вывода	442
Использование функций-“друзей” для перегрузки операторов вывода	443
Перегрузка операторов ввода	445
Сравнение C- и C++-систем ввода-вывода	446
Форматированный ввод-вывод данных	447
Форматирование данных с использованием функций-членов класса <code>ios</code>	447
Установка ширины поля, точности и символов заполнения	451
Использование манипуляторов ввода-вывода	452
Создание собственных манипуляторных функций	454
Файловый ввод-вывод	456
Как открыть и закрыть файл	456
Чтение и запись текстовых файлов	458
Неформатированный ввод-вывод данных в двоичном режиме	460
Использование функций <code>get()</code> и <code>put()</code>	460
Считывание и запись в файл блоков данных	461
Обнаружение конца файла	463
Пример сравнения файлов	464
Использование других функций двоичного ввода-вывода	465
Произвольный доступ	468
Проверка статуса ввода-вывода	470
Использование перегруженных операторов ввода-вывода при работе с файлами	471
Глава 19. Динамическая идентификация типов и операторы приведения типа	473
Динамическая идентификация типов (RTTI)	474
Пример RTTI-приложения	478
Применение оператора <code>typeid</code> к шаблонным классам	480
Операторы приведения типов	483
Оператор <code>dynamic_cast</code>	483
Оператор <code>const_cast</code>	488
Оператор <code>static_cast</code>	489

Оператор <code>reinterpret_cast</code>	490
Сравнение обычной операции приведения типов с новыми четырьмя <code>cast</code> -операторами	491
Глава 20. Пространства имен и другие темы	493
Пространства имен	494
Понятие пространства имен	494
Инструкция <code>using</code>	497
Неименованные пространства имен	499
Пространство имен <code>std</code>	500
Указатели на функции	502
Как найти адрес перегруженной функции	505
Статические члены класса	506
Применение к функциям-членам модификаторов <code>const</code> и <code>mutable</code>	508
Использование <code>explicit</code> -конструкторов	510
Чем интересно неявное преобразование конструктора	511
Синтаксис инициализации членов класса	512
Использование ключевого слова <code>asm</code>	514
Спецификация компоновки	515
Операторы указания на члены “ <code>.</code> ”, “ <code>*</code> ” и “ <code>-></code> ”	516
Создание функций преобразования	519
Глава 21. Введение в стандартную библиотеку шаблонов	521
Обзор STL	522
Контейнерные классы	525
Векторы	526
Доступ к вектору с помощью итератора	531
Вставка и удаление элементов из вектора	532
Сохранение в векторе объектов класса	533
О пользе итераторов	535
Списки	536
Сортировка списка	541
Объединение одного списка с другим	542
Хранение в списке объектов класса	543
Отображения	545
Хранение в отображении объектов класса	549
Алгоритмы	551
Подсчет элементов	554
Удаление и замена элементов	555
Реверсирование последовательности	557
Преобразование последовательности	557
Исследование алгоритмов	559
Класс <code>string</code>	559
Обзор функций-членов класса <code>string</code>	563
Основные манипуляции над строками	563
Поиск в строке	565
Сравнение строк	566
Получение строки с завершающим нулем	566
Хранение строк в других контейнерах	567
И еще об STL	568

Глава 22. Препроцессор C++	569
Директива #define	570
Макроопределения, действующие как функции	572
Директива #error	574
Директива #include	574
Директивы условной компиляции	575
Директивы #if, #else, #elif и #endif	575
Директивы #ifdef и #ifndef	577
Директива #undef	578
Использование оператора defined	579
О роли препроцессора	579
Директива #line	580
Директива #pragma	580
Операторы препроцессора “#” и “##”	581
Зарезервированные макроимена	582
Мысли “под занавес”	582
Приложение А. С-ориентированная система ввода-вывода	583
Использование потоков в С-системе ввода-вывода	585
Функции printf() и scanf()	585
Функция printf()	585
Функция scanf()	588
С-система обработки файлов	592
Функция fopen()	593
Функция fputc()	594
Функция fgetc()	595
Функция feof()	595
Функция fclose()	595
Использование функций fopen(), fgetc(), fputc() и fclose()	596
Функции ferror() и rewind()	597
Функции fread() и fwrite()	597
Функция fseek() и выполнение ввода-вывода с произвольным доступом	599
Функции fprintf() и fscanf()	600
Удаление файлов	600
Приложение Б. Использование устаревшего C++-компилятора	601
Два простых изменения	603
Приложение В. .NET-расширения для C++	605
Ключевые слова .NET-среды	606
__abstract	606
__box	607
__delegate	607
__event	607
__finally	607
__gc	607
__identifier	607
__interface	608
__nogc	608
__pin	608
__property	608

__sealed	608
__try_cast	608
__typeof	609
__value	609
Расширения препроцессора	609
Атрибут attribute	609
Компиляция управляемых C++-программ	609
Предметный указатель	610