

УДК 004.4
ББК 32.973.202
Ч88

Чукич И.

Ч88 Функциональное программирование на языке С++ / пер. с англ. В. Ю. Винника, А. Н. Киселева. – М.: ДМК Пресс, 2020. – 360 с.: ил.

ISBN 978-5-97060-781-7

Язык С++ обычно ассоциируется с объектно-ориентированным программированием. Автор книги доказывает, что на С++ так же удобно создавать программы и в функциональном стиле. Это дает ряд преимуществ, повышая удобство кода и снижая вероятность возникновения ошибок.

Книга разделена на две части. В первой читатель знакомится с азами функционального программирования: основными идиомами и способами их воплощения в языке С++. Вторая часть затрагивает более сложные аспекты и посвящена собственно разработке программ с использованием функционального подхода.

Издание предназначено для опытных разработчиков на С++, желающих расширить границы использования этого языка и повысить эффективность работы.

УДК 004.4
ББК 32.973.202

Original English language edition published by Manning Publications USA, USA. Copyright © 2019 by Manning Publications Co. Russian-language edition copyright © 2020 by DMK Press. All rights reserved.

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

ISBN 978-1-617-29381-8 (англ.)
ISBN 978-5-97060-781-7 (рус.)

Copyright © 2019 by Manning Publications Co.
© Оформление, издание, перевод, ДМК Пресс, 2020

Оглавление

1	■ Введение в функциональное программирование	22
2	■ Первые шаги в функциональном программировании	47
3	■ Функциональные объекты	75
4	■ Средства создания новых функций из имеющихся	107
5	■ Чистота функций: как избежать изменяемого состояния	141
6	■ Ленивые вычисления	167
7	■ Диапазоны	191
8	■ Функциональные структуры данных	209
9	■ Алгебраические типы данных и сопоставление с образцом	226
10	■ Монады	254
11	■ Метапрограммирование на шаблонах	284
12	■ Функциональный дизайн параллельных систем	309
13	■ Тестирование и отладка	338

Содержание

Предисловие	12
Благодарности	14
Об этой книге	15
Об авторе	21

1 Введение в функциональное программирование	22
1.1 Что такое функциональное программирование	23
1.1.1 Соотношение функционального программирования с объектно-ориентированным	25
1.1.2 Сравнение императивной и декларативной парадигм на конкретном примере	25
1.2 Чистые функции	31
1.2.1 Устранение изменяемого состояния	34
1.3 Функциональный стиль мышления	36
1.4 Преимущества функционального программирования	38
1.4.1 Краткость и удобочитаемость кода	39
1.4.2 Параллельная обработка и синхронизация	41
1.4.3 Непрерывная оптимизация	42
1.5 Эволюция C++ как языка функционального программирования	42
1.6 Что узнает читатель из этой книги	44
Итоги	45

2 Первые шаги в функциональном программировании	47
2.1 Функции с аргументами-функциями	48
2.2 Примеры из библиотеки STL	50
2.2.1 Вычисление средних	51

2.2.2	Свертки.....	53
2.2.3	Удаление лишних пробелов.....	58
2.2.4	Разделение коллекции по предикату.....	60
2.2.5	Фильтрация и преобразование.....	62
2.3	Проблема композиции алгоритмов из библиотеки STL.....	64
2.4	Создание своих функций высшего порядка.....	66
2.4.1	Передача функции в качестве аргумента.....	66
2.4.2	Реализация на основе циклов.....	67
2.4.3	Рекурсия и оптимизация хвостового вызова.....	68
2.4.4	Реализация на основе свертки.....	72
	Итоги.....	74

3	Функциональные объекты.....	75
3.1	Функции и функциональные объекты.....	76
3.1.1	Автоматический вывод возвращаемого типа.....	76
3.1.2	Указатели на функции.....	80
3.1.3	Перегрузка операции вызова.....	81
3.1.4	Обобщенные функциональные объекты.....	84
3.2	Лямбда-выражения и замыкания.....	86
3.2.1	Синтаксис лямбда-выражений.....	88
3.2.2	Что находится у лямбда-выражений «под капотом».....	89
3.2.3	Создание лямбда-выражений с произвольными переменными-членами.....	92
3.2.4	Обобщенные лямбда-выражения.....	94
3.3	Как сделать функциональные объекты еще лаконичнее.....	95
3.3.1	Объекты-обертки над операциями в стандартной библиотеке.....	98
3.3.2	Объекты-обертки над операциями в сторонних библиотеках.....	100
3.4	Обертка над функциональными объектами – класс <code>std::function</code>	103
	Итоги.....	105

4	Средства создания новых функций из имеющихся.....	107
4.1	Частичное применение функций.....	108
4.1.1	Универсальный механизм превращения бинарных функций в унарные.....	110
4.1.2	Использование функции <code>std::bind</code> для фиксации значений некоторых аргументов функции.....	114
4.1.3	Перестановка аргументов бинарной функции.....	116
4.1.4	Использование функции <code>std::bind</code> с функциями большего числа аргументов.....	118
4.1.5	Использование лямбда-выражений вместо функции <code>std::bind</code>	121
4.2	Карринг – необычный взгляд на функции.....	124

4.2.1	Простой способ создавать каррированные функции	125
4.2.2	Использование карринга для доступа к базе данных	127
4.2.3	Карринг и частичное применение функций	130
4.3	Композиция функций	132
4.4	Повторное знакомство с подъемом функций	136
4.4.1	Переворачивание пар – элементов списка	138
	Итоги	140

5	Чистота функций: как избежать изменяемого состояния	141
5.1	Проблемы изменяемого состояния	142
5.2	Чистые функции и референциальная прозрачность	145
5.3	Программирование без побочных эффектов	148
5.4	Изменяемые и неизменяемые состояния в параллельных системах	152
5.5	О важности констант	156
5.5.1	Логическая и внутренняя константность	159
5.5.2	Оптимизированные функции-члены для временных объектов	161
5.5.3	Недостатки константных объектов	163
	Итоги	165

6	Ленивые вычисления	167
6.1	Ленивые вычисления в языке C++	168
6.2	Ленивые вычисления как средство оптимизации программ	172
6.2.1	Ленивая сортировка коллекций	172
6.2.2	Отображение элементов в пользовательском интерфейсе	174
6.2.3	Подрезка дерева рекурсивных вызовов за счет запоминания результатов функции	175
6.2.4	Метод динамического программирования как разновидность ленивого вычисления	178
6.3	Универсальная мемоизирующая обертка	180
6.4	Шаблоны выражений и ленивая конкатенация строк	184
6.4.1	Чистота функций и шаблоны выражений	188
	Итоги	190

7	Диапазоны	191
7.1	Введение в диапазоны	193
7.2	Создание представлений данных, доступных только для чтения	194
7.2.1	Функция <i>filter</i> для диапазонов	194

7.2.2	Функция <i>transform</i> для диапазонов	196
7.2.3	Ленивые вычисления с диапазоном значений	197
7.3	Изменение значений с помощью диапазонов	199
7.4	Ограниченные и бесконечные диапазоны	201
7.4.1	Использование ограниченных диапазонов для оптимизации обработки входных диапазонов	201
7.4.2	Создание бесконечного диапазона с помощью ограничителя	203
7.5	Использование диапазонов для вычисления частоты слов	204
Итоги	208

8	Функциональные структуры данных	209
8.1	Неизменяемые связанные списки	210
8.1.1	Добавление и удаление элемента в начале списка	210
8.1.2	Добавление и удаление элемента в конце списка	212
8.1.3	Добавление и удаление элемента в середине списка	213
8.1.4	Управление памятью	213
8.2	Неизменяемые векторы	216
8.2.1	Поиск элементов в префиксном дереве	218
8.2.2	Добавление элементов в конец префиксного дерева	220
8.2.3	Изменение элементов в префиксном дереве	223
8.2.4	Удаление элемента из конца префиксного дерева	223
8.2.5	Другие операции и общая эффективность префиксных деревьев	223
Итоги	225

9	Алгебраические типы данных и сопоставление с образцом	226
9.1	Алгебраические типы данных	227
9.1.1	Определение типов-сумм через наследование	229
9.1.2	Определение типов-сумм с использованием объединений и <i>std::variant</i>	232
9.1.3	Реализация конкретных состояний	235
9.1.4	Особый тип-сумма: необязательные значения	237
9.1.5	Типы-суммы для обработки ошибок	240
9.2	Моделирование предметной области с алгебраическими типами	245
9.2.1	Простейшее решение	246
9.2.2	Более сложное решение: проектирование сверху вниз	247
9.3	Алгебраические типы и сопоставление с образцом	248
9.4	Сопоставление с образцом с помощью библиотеки <i>Mach7</i>	251
Итоги	253

10	Монады	254
10.1	Функторы	255
10.1.1	Обработка необязательных значений	256
10.2	Монады: расширение возможностей функторов	259
10.3	Простые примеры	262
10.4	Генераторы диапазонов и монад	265
10.5	Обработка ошибок	268
10.5.1	<code>std::optional<T></code> как монада	268
10.5.2	<code>expected<T, E></code> как монада	270
10.5.3	Исключения и монады	271
10.6	Обработка состояния с помощью монад	273
10.7	Монады, продолжения и конкурентное выполнение	275
10.7.1	<code>Typ future</code> как монада	277
10.7.2	Реализация типа <code>future</code>	279
10.8	Композиция монад	281
	Итоги	283
11	Метапрограммирование на шаблонах	284
11.1	Манипулирование типами во время компиляции	285
11.1.1	Проверка правильности определения типа	288
11.1.2	Сопоставление с образцом во время компиляции	290
11.1.3	Получение метайнформации о типах	293
11.2	Проверка свойств типа во время компиляции	294
11.3	Каррирование функций	296
11.3.1	Вызов всех вызываемых объектов	299
11.4	Строительные блоки предметно-ориентированного языка	302
	Итоги	307
12	Функциональный дизайн параллельных систем	309
12.1	Модель акторов: мышление в терминах компонентов	310
12.2	Простой источник сообщений	314
12.3	Моделирование реактивных потоков данных в виде монад	318
12.3.1	Создание приемника для сообщений	319
12.3.2	Преобразование реактивных потоков данных	323
12.3.3	Создание потока заданных значений	325
12.3.4	Объединение потоков в один поток	326
12.4	Фильтрация реактивных потоков	327
12.5	Обработка ошибок в реактивных потоках	328
12.6	Возврат ответа клиенту	331
12.7	Создание акторов с изменяемым состоянием	335
12.8	Распределенные системы на основе акторов	336
	Итоги	337

13	Тестирование и отладка	338
13.1	Программа, которая компилируется, – правильная?	339
13.2	Модульное тестирование и чистые функции	341
13.3	Автоматическое генерирование тестов	343
13.3.1	Генерирование тестовых случаев	343
13.3.2	Тестирование на основе свойств	345
13.3.3	Сравнительное тестирование	347
13.4	Тестирование параллельных систем на основе монад	348
	Итоги	352
	Предметный указатель	353