Потопахин В. В.

ИСКУССТВО АЛГОРИТМИЗАЦИИ

ДМК ПРЕСС
ИЗДАТЕЛЬСТВО
Москва, 2011

УДК 004.421
ББК 32.973.26-018
П64

Потопахин В. В.

П64 Искусство алгоритмизации. – М.: ДМК Пресс, 2011. – 320 с.: ил.

ISBN 978-5-94074-621-8

Эта книга для тех, кто хорошо, владея языком программирования и устойчивыми навыками решения задач, желает наработать свой программистский инструментарий. В книге, неформально и довольно детально, разобран значительный набор алгоритмов и методов. Большая часть представленных алгоритмов доведена до реализации на языке Компонентный Паскаль. Для большей прозрачности изложения реализация выполнена пошагово с четкой формулировкой задач каждого шага и записью программного фрагмента. Изложение сопровождается заданиями для самостоятельной работы, количество и сложность которых достаточны для хорошего усвоения материала. Требования к математическим знаниям минимальны, некоторые важные математические понятия и темы кратко изложены в приложении.

К изданию прилагается CD, на котором находится бесплатная среда программирования Блэкбокс, запустив которую, вы сразу сможете начать работу, а также сборник листингов к книге.

УДК 004.421
ББК 32.973.26-018

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.
Материал, изложенный в данной книге, многократно проверен. Но поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-5-94074-621-8

© Потопахин В., 2010
© Оформление, издание, ДМК Пресс, 2011

Содержание

Введение 6

Глава1. Парадигма структурного программирования 9
Зачем нужны общие принципы? 10
Нисходящее проектирование 12
Три базовых элемента структурного программирования 14
Пример разработки 17

Глава 2. Вычислительные алгоритмы 26
Моделирование непрерывных процессов дискретными 27
Метод половинного деления. Общая задача поиска величины 31
Метод касательных 34
Метод хорд 35
Метод итераций (последовательных приближений) 36
Обобщение метода половинного деления 37
Метод наименьших квадратов 38
Задача вычисления площадей криволинейных фигур 42
Метод Симпсона 45
Метод Монте-Карло 48

Глава 3. Числовые алгоритмы 54
Алгоритм Евклида 55
Алгоритмы факторизации и поиска простых 57
Выделение полного квадрата (алгоритм Ферма) 58
Квадратичное решето 60
Алгоритм Полларда 66
Алгоритмы поиска простых чисел 69
Решето Аткина 71
Решето Сундарама 72
Тесты простоты 73
Числа Мерсенна 75
Тест Люка-Лемера 76
Числа Ферма 78
Тест Пепина 78
Псевдослучайные числа 78
Критерии правильности случайных чисел 81
Критерий, основанный на квадратичном отклонении 81
Линейный конгруэнтный метод 81
Методы перемешивания 85

Глава 4. Арифметика 89
Представление числа в позиционной системе счисления 90
Проблемы технической реализации арифметики 93
Двоичный сумматор 94
Ускорение операции сложения 95
Представление чисел в форме с фиксированной и плавающей десятичной точкой 96
Реализация арифметики на уровне алгоритмического языка 97
Сложение двух чисел 97
Вычитание из большего меньшего 99
Умножение 102
Деление 107
Некоторые другие алгоритмы 115
Алгоритм быстрого возведения в степень 115
Быстрый перевод из десятичной в двоичную систему счисления 116
Решение диофантовых уравнений 117
Двоичная арифметика 119
Сложение двоичных чисел 120
Как преобразовать в двоичное число дробную часть 122
Вычитание двоичных чисел 124
Умножение в двоичной системе счисления 125
Деление в двоичной системе счисления 126

Глава 5. Рекурсия и динамическое программирование 131
Общее определение 132
Задача о ханойской башне 135
Переход от рекурсивного к нерекурсивному решению 138
Рекурсия как метод поиска 143
Динамическое программирование 144
Задача обхода конем шахматной доски 146
Факторизация числа 153

Глава 6. Сортировки 166
Общая постановка задачи 167
Обменные сортировки. Сортировка пузырьком 168
Шейкерная сортировка 170
Анализ качеств алгоритма 171
Сортировка выбором 174
Сортировка вставками 176
Сортировка Шелла 178
Быстрая сортировка 181
Двоичная сортировка 186
Сортировка слияниями 191

Глава 7. Комбинаторные задачи 204
Общая постановка задачи 205
Оптимизация перебора 207
Связь комбинаторики с алгоритмами на графах 209
Основные комбинаторные задачи 210
Задача получения перестановок на множестве из N элементов 210
Построение сочетаний без повторений на множестве элементов 216
Сочетания с повторениями 221
Задача получения размещений 223

Глава 8. Динамические структуры данных 224
Понятие о динамической величине 225
Линейный связный список 226
Зачем рекурсивные структуры нужны? 229
Использование рекурсивных определений для создания деревьев данных 233

Глава 9. Алгоритмы принятия решений 237
Постановка задачи. Понятие эвристического алгоритма 238
Оценочная функция 240
Метод минимакса 241
Альфа-бета алгоритм 245

Глава 10. Алгоритмы на графах 250
Стратегии обхода 251
Обход графа в ширину 251
Обход графа в глубину 253
Построение остовного дерева 253
Алгоритм Прима 254
Алгоритм Краскала 258
Алгоритм поиска компонент связности 263
Волновой алгоритм 265
Алгоритм Дейкстры 269
Алгоритм Флойда 276
Нахождение максимального потока 280

Глава 11. Приложения 296
Приложение 1. Элементы комбинаторики 297
Приложение 2. Теория графов 301
Приложение 3. Элементы теории вероятности 309
Приложение 4. Синтаксис языка Компонентный Паскаль 315

Список литературы 319

Введение

Книга, которую вы держите в руках, является логическим продолжением книги «Современное программирование с нуля» того же автора. Но если упомянутая книга была посвящена выработке базовых программистских умений, то сейчас цель – наработка инструментария программиста-профессионала. Если вы в программировании совсем новичок, то придется эту книгу пока отложить и заняться приобретением базовых навыков: необходимо уверенно владеть языком Паскаль, желательно его последней версией Компонентный Паскаль, и совершенно необходим хороший навык написания хотя бы несложных программ.

Основное содержание книги – алгоритмы и некоторые интересные задачи. Исключение составляет только первая глава, посвященная принципиальному вопросу: что такое хорошо написанная программа. Это, может быть, покажется неинтересным, но постарайтесь все же первую главу прочитать максимально внимательно. Структурное программирование, которому полностью посвящена первая глава, есть форма дисциплины мышления программиста. А недисциплинированный программист обречен на неудачу независимо от того, каким набором алгоритмов, технологий и языков программирования он владеет.

Все остальные главы посвящены искусству алгоритмизации. Часть материала потребует некоторых математических знаний. Большую часть требуемой математики вы сможете найти здесь же, но без строгих доказательств и детального изложения. Поэтому книга довольно самодостаточна, а для желающих углубить свои знания по тому или иному вопросу даны ссылки на специальные источники.

Язык изложения – Компонентный Паскаль. Для примеров практически не используются какие-либо библиотечные модули, применяемые средства максимально просты, если вы имеете хороший языковой опыт, однако не знаете КП, текст не станет для вас слишком непонятным. Но все же книга будет более читаемой, если вы как следует усвоите Компонентный Паскаль.

Наверное, главная особенность стиля изложения – это детальность разработки примеров. Только некоторые совсем уж простые задачи даны кратко, большая их часть снабжена пошаговым разъяснением всех деталей реализации. Если алгоритм сложен, то его объяснение снабжается примерами использования, иллюстрациями. Выбор реализации алгоритмов сделан автором в пользу прозрачности и понятности, быть может, иногда за счет потери некоторой части эффективности. Уровень завершенности реализации учебных примеров разный. Для некоторых примеров дан только фрагмент программы, но таких мало. Для некоторых примеров написан текст процедуры, которую еще надо оформить в какой-то модуль. И есть примеры, полностью завершенные (до модуля). Впрочем, если текст примера в книге дан только в виде процедуры, то в приложении на диске он скорее всего представляет собой полностью завершенную программу. В текстах примеров активно используются идентификаторы на русском языке для большей эффективности объяснения. Поэтому если у вас нет желания переписывать иденти-

фикаторы латиницей, то воспользуйтесь сборкой BlackBox с диска, прилагаемого к книге.

В тексте книги много заданий для самостоятельной работы. Все задания логически вытекают из хода изложения. Это могут быть теоретические вопросы о свойствах исследуемых алгоритмов, это могут быть предложения по улучшению реализации или идеи несколько иной реализации того же алгоритма. Немного пройдемся по главам.

Первая глава посвящена основным идеям структурного программирования. Глава очень краткая, изложена, если можно так сказать, самая суть метода. Здесь необходимо указать, что вопросы методологии всегда были и будут самыми спорными, поэтому, возможно, кто-то не согласен с такой структурой рассказа, очевидно, что вопросы структурного программирования и нисходящего проектирования можно излагать по-разному. Но перед книгой не ставилась цель исчерпывающего анализа этой сложнейшей темы, кроме того, в тексте главы будут ссылки на авторитетных авторов и классические книги.

Вторая глава – о вычислительных алгоритмах, это о том, как поступать в ситуациях, когда нет возможности выполнить расчет подстановкой в простую формулу. Оказывается, в реальных задачах очень часто приходится прибегать к так называемым численным методам, которые и составляют содержание второй главы.

Третья глава – это рассказ о числовых алгоритмах. Объяснены некоторые часто используемые вещи, например алгоритм Евклида, решето Эратосфена, и часть времени посвящена достаточно увлекательным задачам, не имеющим на сегодня исчерпывающего решения, – это задача факторизации, задача получения больших простых, задача построения последовательности псевдопростых чисел.

Четвертая глава – это арифметика. Мы все привыкли, что электронные устройства умеют выполнять арифметические операции, но ведь это тоже проблемы алгоритмизации. В главе рассмотрены некоторые вопросы, возникающие при программировании арифметики. Приведены реализации выполнения операций столбиком, и рассказано о некоторых возможностях усиления арифметических алгоритмов.

Пятая глава – рассказ о рекурсии. Даны определение и основные свойства. Рассказано, как строится рекурсивный процесс, какие при этом возникают проблемы. Вводится представление о динамическом программировании. Решены несколько несложных задач, и завершается глава двумя достаточно серьезными задачами: задачей обхода конем шахматной доски и еще одним алгоритмом факторизации, которого нет в главе о числовых алгоритмах.

Шестая глава. Сортировки. Рассмотрены: пузырьковая сортировка, шейкерная, сортировка подсчетом, сортировка вставками, выбором, быстрая, двоичная, сортировка Шелла, сортировка слияниями и естественными слияниями. Начинается глава общей постановкой задачи, в ходе изложения кратко анализируются свойства сортировок.

Седьмая глава. Комбинаторные задачи. Дано представление о том, что такое вообще комбинаторная задача, обсуждены проблема комбинаторного взрыва и возможности построения эвристического решения. Приведены реализации не-

которых базовых, комбинаторных задач: построение перестановок, сочетаний, с повторениями и без. Даны рекурсивные и нерекурсивные решения.

Восьмая глава. Динамические структуры данных. Это небольшой рассказ о том, что такое динамические величины и, главным образом, что такое рекурсивно определяемые величины, описаны некоторые операции над связными списками и деревьями.

Девятая глава – о том, как написать программу, умеющую принимать решения. Разъяснены основные составляющие такой программы: оценочная функция, минимаксный обход дерева вариантов, некоторые возможности его сокращения. Дано определение эвристического алгоритма, указаны проблемы, возникающие при попытке сократить дерево перебора.

Последняя, десятая глава – самая технически сложная – алгоритмы на графах. Рассмотрены следующие задачи: построение остовного дерева, построение компоненты связности, поиск кратчайшего пути волновым алгоритмом, поиск наиболее дешевых путей алгоритмами Дейкстры и Флойда, построение максимального потока алгоритмом Форда-Фалкерсона.

Завершена книга несколькими приложениями, кратко излагающими основные понятия комбинаторики, теории графов и теории вероятностей. Это на тот случай, если вашей математической подготовки окажется недостаточно.

Глава 1

Парадигма структурного программирования

Зачем нужны общие принципы.....10
Нисходящее проектирование.....12
Три базовых элемента структурного программирования..... 14
Пример разработки..... 17

Зачем нужны общие принципы?

Попробуем понять, что такое хорошая программа. Очевидно, этот вопрос надо решить дважды: во-первых, с точки зрения пользователя, так как программа в конечном итоге делается для него, и во-вторых, с точки зрения программиста, так как он, наверное, не может быть безразличен к качеству своего продукта. Пользователь программного обеспечения ожидает, что результат будет получен гарантированно и за приемлемое время. Еще одно, важное требование – это стоимость ПО. Если пользователь платит за программу, то естественно, он желает, чтобы стоимость была по возможности низкой. Это почти все. Различные программистские вопросы, вроде того, на каком языке написана программа, как она устроена, сколько в её разработку вложено программистской энергии, его не интересуют.

> ***Важное замечание.*** Есть один исключительно важный момент, выпадающий из логики дальнейшего изложения, но пропустить который нельзя. Это вопрос надежности ПО. Грубо говоря, нажимая на кнопку, мы ожидаем вполне определенный результат, а не какой-нибудь. Иногда это вопрос больших материальных потерь и даже человеческих жизней. И надежность обеспечивается отнюдь не талантом программиста, а скорее выбором и строгим соблюдением общих принципов и правил.

Позиция программиста в вопросе оценки качества программы куда более сложная. Например, программист желает получить заданный результат с минимальными усилиями. Отчасти из такого желания и возникает понятие технологии. В промышленности из желания делать много малыми усилиями возникли разделение труда и конвейер. В программировании результатом такого желания можно считать понятие процедуры, коллекционирование процедур в библиотеки и сборку новой программы из уже готовых процедур. С этой точки зрения задачу программирования можно сформулировать так:

> ***Определите, какие процедуры необходимы, и соберите из них новую процедуру, реализующую поставленную задачу.***

Так звучит парадигма процедурного программирования. Конечно же компоновать программу из процедур и языковых конструкций можно по-разному. Процедурная парадигма ничего не говорит о том, как организовать процесс программирования, она лишь определяет процедуру главным строительным блоком. Вопрос, что такое хорошо и что такое плохо, с этой позиции остается открытым.

Анализ с точки зрения функциональности программы мы опустим, это можно отнести к ожиданиям пользователя, разговор о которых уже закрыт. С точки же зрения программиста существуют еще два важных фактора: скорость написания программы и её читаемость. Читаемость – собственно то же фактор скорости, скорости понимания программы.

Конечно, лучше всего было бы быстро писать и легко читать. Но, к сожалению, в реальном программировании приходится искать золотую середину, отдавая предпочтение либо скорости написания, либо читаемости. Что важнее? На первый взгляд может показаться, что скорость важнее. Чем быстрее мы напишем программу, тем меньше потратим времени и энергии. Но это только на первый