Никлаус Вирт

Алгоритмы и структуры данных

Новая версия для Оберона + CD

Перевод с английского под редакцией
доктора физ-мат. наук, Ткачева Ф. В.

ДМК
издательство

УДК 32.973.26-018.2
ББК 004.438
В52

Никлаус Вирт
В52 Алгоритмы и структуры данных. Новая версия для Оберона + CD / Пер. с англ. Ткачев Ф. В. – М.: ДМК Пресс. – 272 с.: ил.

В классическом учебнике тьюринговского лауреата Н.Вирта аккуратно, на тщательно подобранных примерах прорабатываются основные темы алгоритмики – сортировка и поиск, рекурсия, динамические структуры данных.

Перевод на русский язык выполнен заново, все рассуждения и программы проверены и исправлены, часть примеров по согласованию с автором переработана с целью максимального прояснения их логики (в том числе за счет использования цикла Дейкстры). Нотацией примеров теперь служит Оберон/Компонентный Паскаль – наиболее совершенный потомок старого Паскаля по прямой линии.

Все программы проверены и работают в популярном варианте Оберона – системе Блэкбокс, и доступны в исходниках на прилагаемом CD вместе с самой системой и дополнительными материалами.

Большая часть материала книги составляет необходимый минимум знаний по алгоритмике не только для программистов-профессионалов, но и любых других специалистов, активно использующих программирование в работе.

Книга может быть использована как учебное пособие при обучении будущих программистов, начиная со старшеклассников в профильном обучении, а также подходит для систематического самообразования.

Содержание компакт-диска:

Базовая конфигурация системы Блэкбокс с коллекцией модулей, реализующих программы из книги.
Базовые инструкции по работе в системе Блэкбокс.
Полный перевод документации системы Блэкбокс на русский язык.
Конфигурация системы Блэкбокс для использования во вводных курсах программирования в университетах.
Конфигурация системы Блэкбокс для использования в школах (полная русификация меню, сообщений компилятора, с возможностью использования ключевых слов на русском и других национальных языках).
Доклады участников проекта Информатика-21 по опыту использования системы Блэкбокс в обучении программированию.
Оригинальные дистрибутивы системы Блэкбокс 1.5 (основной рабочий) и 1.6rc6.
Инструкции по работе в Блэкбоксе под Linux/Wine.
Дистрибутив оптимизирующего компилятора XDS Oberon (версии Linux и MS Windows).
OberonScript – аналог JavaScript для использования в Web-приложениях.

ISBN 0-13-022005-9 (анг.)
ISBN 978-5-51967-298-6

© N. Wirth, 1985 (Oberon version: August 2004).
© Перевод на русский язык, исправления и изменения, Ф. В. Ткачев, 2010.
© Оформление, издание, ДМК Пресс, 2010

Содержание

О новой версии классического учебника Никлауса Вирта 5

Предисловие 11

Предисловие к изданию 1985 года 15

Нотация 16

Глава 1. Фундаментальные структуры данных 11
1.1. Введение 18
1.2. Понятие типа данных 20
1.3. Стандартные примитивные типы 22
1.4. Массивы 26
1.5. Записи 29
1.6. Представление массивов, записей и множеств 31
1.7. Файлы или последовательности 35
1.8. Поиск 49
1.9. Поиск образца в тексте (string search) 54
Упражнения 65
Литература 67

Глава 2. Сортировка 69
2.1. Введение 70
2.2. Сортировка массивов 72
2.3. Эффективные методы сортировки 81
2.4. Сортировка последовательностей 97
Упражнения 128
Литература 130

Глава 3. Рекурсивные алгоритмы 131
3.1. Введение 132
3.2. Когда не следует использовать рекурсию 134
3.3. Два примера рекурсивных программ 137
3.4. Алгоритмы с возвратом 143
3.5. Задача о восьми ферзях 149

3.6. Задача о стабильных браках 154
3.7. Задача оптимального выбора 160
Упражнения 164
Литература 166

Глава 4. Динамические структуры данных 167
4.1. Рекурсивные типы данных 168
4.2. Указатели 170
4.3. Линейные списки 175
4.4. Деревья 191
4.5. Сбалансированные деревья 210
4.6. Оптимальные деревья поиска 220
4.7. Б-деревья (B-trees) 227
4.8. Приоритетные деревья поиска 246
Упражнения 250
Литература 254

Глава 5. Хэширование 255
5.1. Введение 256
5.2. Выбор хэш-функции 257
5.3. Разрешение коллизий 257
5.4. Анализ хэширования 261
Упражнения 263
Литература 264

Приложение A. Множество символов ASCII 265

Приложение B. Синтаксис Оберона 266

Приложение C. Цикл Дейкстры 269

О новой версии классического учебника Никлауса Вирта

Новая версия учебника Н. Вирта «Алгоритмы и структуры данных» отличается от английского прототипа [1] сильнее, чем просто исправлением многочисленных опечаток и огрехов, накопившихся в процессе тридцатилетней эволюции книги. Объясняется это целями автора и переводчика при работе над книгой в контексте проекта «Информатика-21» [2], который, опираясь на обширный совокупный опыт ряда высококвалифицированных специалистов (см. списки консультантов и участников на сайте проекта [2]), ставит задачу создания единой системы вводных курсов информатики и программирования, охватывающей учащихся примерно от 5-го класса общей средней школы по 3-й курс университета. Такая система должна иметь образцом и дополнять уникальную российскую систему математического образования. Это предполагает наличие стержня общих курсов, составляющих единство без внутренних технологических барьеров (которые приводят, среди прочего, к недопустимым потерям дефицитного учебного времени) и лишь варьирующихся в зависимости от специализации, вместе с надстройкой из профессионально ориентированных курсов, опирающихся на этот стержень в отношении базовых знаний учащихся. Такая система подразумевает наличие качественных учебников (первым из которых имеет шанс стать данная книга), «говорящих» на общем образцовом языке программирования. Естественный кандидат на роль такого общего языка – Оберон/Компонентный Паскаль. Подробней об Обероне речь пойдет ниже, здесь только скажем, что Паскаль (использованный в первом издании данной книги 1975 г.), Модулу-2 (использованную во втором издании, переведенном на русский язык в 1989 г. [3]) и Оберон (использованный в данной версии) логично рассматривать соответственно как альфа-, бета- и окончательную версию одного и того же языка. Использование Оберона – самое очевидное отличие данной версии книги от предыдущего издания.

В контекст идеи о единой системе вводных курсов вписывается и узкая задача, решавшаяся новой версией учебника, – дать небольшое продуманное пособие, в котором аккуратно, но не топя читателя в болоте второстепенных деталей, прорабатывались бы традиционные темы классической алгоритмики, для полного обсуждения которых нет времени в спецкурсе, читаемом переводчиком с 2001 г. на физфаке МГУ в попытке обеспечить хотя бы минимум культуры программирования у будущих аспирантов. Здесь требуется «отлаженный» текст, пригодный для самостоятельной работы студентов. С точки зрения содержания, лучшим кандидатом на эту роль оказался прототип [1].

Что двойное переделывание программ и рассуждений в тексте (с Паскаля на Модулу-2 и затем на Оберон) не прошло безнаказанно, само по себе неудивительно. Однако затруднения, возникшие при верификации программ и текста, хотя и были преодолены, все же показались чрезмерными. Поэтому, и ввиду учебного назначения книги, встал ребром вопрос о необходимости доработки примеров. Предложения переводчика были одобрены автором на совместной рабочей сессии в апреле сего года и реализованы непосредственно в данном переводе (при первой возможности соответствующие изменения будут внесены и в прототип [1]).

Во-первых, алгоритмы поиска образца в тексте переписаны в терминах цикла Дейкстры (многоветочный while [4]). Эта фундаментальная и мощная управляющая структура поразительным образом до сих пор не представлена в распространенных языках программирования, поэтому ей посвящено новое приложение C. Раздел 1.9, в который теперь выделены эти алгоритмы, будет неплохой иллюстрацией реального применения цикла Дейкстры. Вторая группа заметно измененных программ – алгоритмы с возвратом в главе 3, в которых теперь эксплицировано применение линейного поиска и, благодаря этому, тривиализована верификация. Такое прояснение рекурсивных комбинаторных алгоритмов является довольно общим. Обсуждались – но были признаны в данный момент нецелесообразными – модификации и некоторых других программ.

Надо заметить, что программистский стиль автора вырабатывался с конца 1950-х гг., когда проблема эффективности программ висела над головами программистов дамокловым мечом, и за несколько лет до того, как Дейкстра опубликовал систематический метод построения программ [4]. В старых версиях книги заметна рефлекторная склонность к оптимизации до полного прояснения логики программ, что затрудняло эффективное применение формальной техники. Это легко объяснить: Н. Вирт осваивал только еще формирующиеся систематические методы, непосредственно участвуя в процессе создания программирования как академической дисциплины, версия за версией улучшая свои учебники.

Но и через четверть века после последней существенной переделки учебника автором аналогичная склонность к преждевременной оптимизации при не просто не вполне уверенной, а напрочь отсутствующей формальной технике – и, как следствие, запутанные циклы, – характерные черты стиля «широких программистских масс»! В профессиональных интернет-форумах до сих пор можно найти позорные дискуссии о том, нужно ли учиться писать циклы по Дейкстре, – и это в лучшем случае. Если же вообразить себе весь окружающий нас непрерывно растущий массив софта, от которого наша жизнь зависит все больше, то впору впасть в депрессию: *Quo usque tandem, Catilina*? – Сколько еще нужно *десятилетий*, чтобы система образования вышла, наконец, на уровень, давным-давно достигнутый наукой? Во всяком случае, ясно, что едва ли не главная причина проблемы – хаос, царящий в системе ИТ-образования, тормозящий создание и распространение качественных методик и поддерживаемый, среди прочего, корыстными интересами «монстров» индустрии.

Здесь уместно сказать о языке Оберон/Компонентный Паскаль, пропагандируемом в качестве общей платформы для предполагаемой единой системы курсов

программирования. Оберон – последний большой проект Никлауса Вирта, выдающегося инженера, ученого и педагога, вместе с Бэкусом, А. Ершовым, Дейкстрой, Хоором и другими пионерами компьютерной информатики превратившего программирование в систематическую дисциплину и лучше всего известного созданием серии все более совершенных языков программирования – Паскаля (1970), Модулы-2 (1980) и наконец Оберона (1988, 2007). В этих языках отражалось все более полное понимание проблематики эффективного программирования. Языки эти сохраняют идейную и стилевую преемственность, и коммерсант, озабоченный сохранением доли рынка, не назвал бы их по-разному (ср. зоопарк бейсиков). Чтобы подчеркнуть эту преемственность, самому популярному диалекту Оберона было возвращено законное фамильное имя – Компонентный Паскаль.

Оберон/Компонентный Паскаль унаследовал лучшие черты старого доброго Паскаля и добавил к ним промышленный опыт Модулы-2 (на которой программируются, например, российские спутники связи [5]), а также выверенный минимум средств объектно-ориентированного программирования. Принципальное достижение – удалось наконец добиться герметичности системы типов (теперь ее нельзя обойти средствами языка даже при работе с указателями). Это обеспечило возможность автоматического управления памятью (сбора мусора; до Оберона сбор мусора оставался прерогативой динамических языков – функциональных, скриптовых и т. п.) В результате диапазон эффективного применения Оберона, похоже, шире, чем у любого другого языка: это и вычислительные приложения, и системы управления любого масштаба (от беспилотников весом в 1 кг до грандиозных каскадов ГЭС), и, например, задачи символической алгебры с предельно динамичными структурами данных.

Особо следует остановиться на минимализме Оберона. Традиционно разработчики сосредоточиваются на том, чтобы снабдить свои языки, программы, библиотеки «богатым набором средств» – ведь так легче привлечь клиента, надеющегося побыстрее найти готовое решение для своих прикладных нужд. Погоня за «богатым набором средств» оборачивается ущербом качеству и надежности системы. Вместе с коммерческими соображениями это приводит к тому, что получается большая закрытая сложная система с вроде бы богатым набором средств, но хромающей надежностью и ограниченной расширяемостью, так что если пользователь сталкивается с нестандартной ситуацией в своих приложениях (что случается сплошь и рядом – ведь разнообразие реального мира превосходит любое воображение писателей библиотек), то он оказывается в тупике.

Н. Вирт еще со времен Паскаля, созданного в пику фантазийному Алголу-68 [6], пошел другим путем. Его гамбит заключался в том, чтобы, отказавшись от включения в язык *максимума* средств на все случаи жизни, тщательнейшим образом выделить *минимум* реально ключевых средств, – обязательно включив в этот минимум все, что нужно для безболезненной, неограниченной расширяемости программных систем, – и добиться высоконадежной реализации такого ядра. Этот замысел был с блеском реализован Н. Виртом и его соратником Ю. Гуткнехтом в проекте Оберон [7]. Минимализм и уникальная надежность Оберона

заставляют вспомнить автомат Калашникова. При этом вся мощь Оберона оказывается открытой даже программистам-непрофессионалам – физикам, инженерам, лингвистам.., занятым программированием изрядную долю своего рабочего времени.

Для преподавателя важно, что в Обероне достигнуты ортогональность и свободная комбинируемость языковых средств, смысловая прозрачность, а также беспрецедентно малый для столь мощного языка размер (см. полное описание синтаксиса в приложении В, а также обсуждение в [8]). В этом отношении Оберон побеждает за явным преимуществом традиционные промышленные языки, пресловутая избыточная сложность которых оказывается источником своего рода ренты, взимаемой с остального мира. Оберон скромно уходит в тень при рассмотрении любой языково-неспецифичной темы – от введения в алгоритмику до принципов компиляции и программной архитектуры. А после постановки базовой техники программирования на Обероне изучение промышленных языков зачастую сводится к изучению способов обходить дефекты их дизайна. Если уже старый Паскаль оказался настолько удачной платформой для обучения программированию, что принес своему автору высшую почесть в компьютерной информатике – премию им. Тьюринга, то понятно, что буквально вылизанный Оберон/Компонентный Паскаль называют уже «практически идеальной» платформой для обучения программированию.

Имея в виду исключительные педагогические достоинства Оберона, для всех примеров программ, приведенные в книге, обеспечена воспроизводимость в системе программирования для Компонентного Паскаля, известной как Блэкбокс (BlackBox Component Builder [9]). Это пулярный вариант Оберона, созданный для работы в распространенных операционных системах. Конфигурации Блэкбокса для использования в школе и университете доступны на сайте проекта «Информатика-21» [2]. Открытый, бесплатный и безупречно современный Блэкбокс оказывается естественной заменой устаревшему Турбо Паскалю – заменой тем более привлекательной, что, несмотря на минимализм и благодаря автоматическому управлению памятью, это более мощный инструмент, чем промышленные системы программирования на диалектах старого Паскаля. Краткое описание возможностей Блэкбокса с точки зрения использования в школьных курсах можно найти в статье [10].

Важное приложение к книге – полный комплект программ, представленных в тексте учебника, в виде, готовом к выполнению. Программы оформлены в отдельных модулях вместе с необходимыми вспомогательными процедурами, и все такие модули собраны в папке ADru/Mod/, которая должна лежать внутри основной папки Блэкбокса (следует иметь в виду, что файлы с расширением *.odc должны читаться из Блэкбокса). Читатель без труда разберется с компиляцией и запуском программ по комментариям в модулях, читая модули в том порядке, в каком они встречаются в тексте книги (или в лексикографическом порядке имен файлов). В тексте книги в начальных строках каждого законченного программного примера справа указано имя соответствующего модуля. Например, комментарий (*ADruS18_Поиск*) означает, что данная программа содержится в модуле

ADruS18_Поиск, который в соответствии с правилами Блэкбокса хранится в файле ADru/Mod/S18_Поиск.odc. При этом речь идет о программе из раздела 1.8, а необязательный суффикс "_Поиск" служит удобству ориентации. Вся папка ADru в составе Блэкбокса имеется на диске, если диск приложен к книге, либо может быть скачана с адреса [11].

Наконец, несколько слов о собственно переводе. Старый перевод [3] был выполнен, что называется, из общих соображений. Но совсем другое дело – иметь в виду конкретных студентов, не обязательно будущих профессиональных программистов, пытающихся за минимальное время овладеть основами программирования. Поэтому в новом переводе были предприняты особые усилия, чтобы избежать размывания смысла из-за неточностей, неизбежно вкрадывающихся при неполном понимании переводчиком оригинала (ср. примечание на с. 110 в главе о сортировках в [3], где выражена надежда, что «сам читатель разберется, что хотел сказать автор»). Например, при более-менее прямолинейной пофразовой интерпретации малейшая неточность способна развалить смысл лаконичного текста Вирта из-за того, например, что после перевода могут перестать быть однокоренными слова, благодаря которым только и обеспечивалась смысловая связь между предложениями в оригинале. Поэтому добиться полного сохранения смысла при переводе оказалось проще, выполнив его с нуля.

В отношении терминологии переводам специалистов было отдано должное. Вслед за Д. Б. Подшиваловым [3] мы используем прилагательные «массивовый», «последовательностный» и «записевый». Решающий довод в пользу таких прилагательных – они естественно вписываются в грамматическую систему русского языка, чем обеспечивается необходимая гибкость выражения.

Однако даже в отношении терминологии переводы по компьютерной тематике часто демонстрируют неполное понимание существенных деталей английской грамматики. Например, при использовании существительного в качестве определения в препозиции (что, кстати, не эквивалентно русской конструкции, выражаемой родительным падежом) множественное число может нейтрализоваться, и при переводе на русский его иногда нужно восстанавливать. Так, path length должно переводиться не как «длина пути», а как «длина путей», что, между прочим, прямо соответствует математическому определению и ощутимо помогает понимать рассуждения. Optimal search tree – «оптимальное дерево поиска», а не «дерево оптимального поиска». Advanced sort algorithms – «эффективные алгоритмы сортировки», потому что буквальное значение advanced в данном случае давно нейтрализовано. Переводить на русский язык двумя словами специфичные для стилистики английского языка синонимичные пары вроде «methods and techniques» обычно неразумно. И так далее. Масса подобных неточностей снижает удобочитаемость текста и затемняет и без того непростой смысл оригинала.

Хотя по конкретным стилистическим вопросам копья можно ломать до бесконечности, все же хочется надеяться, что предпринятые усилия в основном достигли цели – не потерять точный смысл английского «исходника» этого выдержавшего проверку временем прекрасного учебника.

Троицк, Московская обл., июль 2009 Ф. В. Ткачев

[1] Wirth N. Algorithms and Data Structures. Oberon version: 2004 //http://www.inr.ac.ru/~info21/pdf/AD.pdf
[2] Информатика-21: Международный общественный научно-образовательный проект // http://www.inr.ac.ru/~info21/
[3] Н. Вирт. Алгоритмы и структуры данных / пер. с англ. Д. Б. Подшивалова. – М.: Мир, 1989.
[4] Дейкстра Э. Дисциплина программирования. – М.: Мир, 1978.
[5] Koltashev A. A., in: Lecture Notes in Computer Science 2789. – Springer-Verlag, 2003.
[6] Кто такой Никлаус Вирт? // http://www.inr.ac.ru/~info21/wirth/wirth.htm
[7] Wirth N. and Gutknecht J. Project Oberon. – Addison-Wesley, 1992.
[8] Свердлов С. В. Языки программирования и методы трансляции. – СПб.: Питер, 2007.
[9] http://www.oberon.ch/blackbox.html
[10] Ильин А. С. и Попков А. И. Компонентный Паскаль в школьном курсе информатики // http://inf.1september.ru/article.php?ID=200800100
[11] http://www.inr.ac.ru/~info21/ADru/