

Министерство образования и науки РФ  
ФГБОУ ВПО «Тульский государственный педагогический университет  
имени Л. Н. Толстого»

**Г. В. Ваныкина, Т. О. Сундукова**

# **АЛГОРИТМЫ КОМПЬЮТЕРНОЙ ОБРАБОТКИ ДАННЫХ**

*Учебное пособие*

*Рекомендовано УМО  
в области инновационных междисциплинарных  
образовательных программ в качестве учебного пособия  
по направлению 010500 «Математическое обеспечение  
и администрирование информационных систем»*

Тула  
Издательство ТГПУ им. Л.Н. Толстого  
2011

УДК 004.421  
ББК 32.973.26-018.2  
В17

*Рецензенты:*

доктор педагогических наук, профессор *А. Р. Есаян*  
(кафедра информатики и методики обучения информатике  
ТГПУ им. Л. Н. Толстого);  
кандидат физико-математических наук, доцент *И. Ю. Баженова*  
(факультет ВМК МГУ им. М. В. Ломоносова)

**Ваныкина, Г. В., Сундукова, Т. О.**

В17 Алгоритмы компьютерной обработки данных: Учеб. пособие /  
Г. В. Ваныкина, Т. О. Сундукова. – Тула: Изд-во Тул. гос. пед. ун-та  
им. Л. Н. Толстого, 2011. – 219 с.

ISBN 978-5-87954-650-7

Пособие представляет собой подробное изложение алгоритмов компьютерной обработки структурированных типов данных. Для изучения и реализации алгоритмов решения прикладных задач на языке C++, учета технологических особенностей конкретной среды исполнения предложен комплекс тематических разделов. Каждый раздел содержит необходимый теоретический и справочный материал, большое количество примеров программных кодов с комментариями, задания для аудиторной и/или самостоятельной работы.

Данное пособие обобщает и систематизирует различные абстракции данных, поддерживаемые в C++, в контексте применяемых для их обработки алгоритмов.

Предназначено для студентов специальности 351500 «Математическое обеспечение и администрирование информационных систем», направлений подготовки 010500.62 «Математическое обеспечение и администрирование информационных систем», 010300.62 «Фундаментальная информатика и информационные технологии». Отдельные разделы могут быть использованы при обучении программированию студентов специальности 030100 «Информатика» и направлений подготовки 540200 «Физико-математическое образование» (профиль 540203 «Информатика»), 050100.62 «Педагогическое образование» (профиль «Информатика»). Может быть использовано студентами и преподавателями вузов, средних профессиональных и средних общеобразовательных учреждений.

**ББК 32.973.26-018.2**

**УДК 004.421**

ISBN 978-5-87954-650-7

© Г. В. Ваныкина, Т. О. Сундукова, 2011  
© ТГПУ им. Л. Н. Толстого, 2011

# СОДЕРЖАНИЕ

<b>Предисловие .....</b>	<b>7</b>
<b>1. Алгоритмы обработки данных .....</b>	<b>9</b>
1.1. Понятие «алгоритм обработки данных» .....	9
1.2. Ресурсная эффективность алгоритмов .....	10
1.3. Методы оценки ресурсной эффективности алгоритмов .....	13
1.4. Базовые алгоритмы обработки данных .....	16
Ключевые термины .....	17
Краткие итоги .....	18
Материалы для практики .....	18
Литература .....	19
<b>2. Рекурсия и рекурсивные алгоритмы .....</b>	<b>20</b>
2.1. Основные понятия рекурсии .....	20
2.2. Анализ трудоемкости рекурсивных алгоритмов методом подсчета вершин дерева рекурсии .....	22
2.3. Примеры разработки рекурсивной триады .....	23
Ключевые термины .....	31
Краткие итоги .....	32
Материалы для практики .....	32
Литература .....	33
<b>3. Решение задач на использование рекурсивных алгоритмов .....</b>	<b>34</b>
3.1. Основные методы решения задач с помощью рекурсии .....	34
3.2. Опорная схема «Увидеть» .....	35
3.3. Опорная схема «Переформулировать» .....	35
3.4. Опорная схема «Обобщить» .....	36
3.5. Опорная схема «Характеристические свойства» .....	37
3.6. Опорная схема «Перенести часть условий в проверку» .....	39
3.7. Опорная схема «Обратить функцию» .....	39
3.8. Опорная схема «Найти родственника» .....	40
Ключевые термины .....	41
Краткие итоги .....	42
Материалы для практики .....	43
Литература .....	44
<b>4. Алгоритм перебора с возвратом .....</b>	<b>45</b>
4.1. Перебор с возвратом .....	45
4.2. Вычислительная схема перебора с возвратом .....	46
4.3. Использование перебора с возвратом при решении задач .....	47
Ключевые термины .....	53
Краткие итоги .....	54
Материалы для практики .....	54
Литература .....	55

<b>5. Алгоритмы поиска в линейных структурах.....</b>	<b>56</b>
5.1. Основные понятия и общий алгоритм поиска данных .....	56
5.2. Последовательный (линейный) поиск .....	57
5.3. Бинарный (двоичный) поиск.....	58
Ключевые термины .....	60
Краткие итоги .....	61
Материалы для практики.....	61
Литература .....	64
<b>6. Алгоритмы хеширования данных .....</b>	<b>65</b>
6.1. Основные понятия хеширования данных.....	65
6.2. Методы разрешения коллизий.....	67
6.3. Алгоритмы хеширования .....	69
6.3.1. Таблица прямого доступа .....	69
6.3.2. Метод остатков от деления.....	70
6.3.3. Метод функции середины квадрата.....	70
6.3.4. Метод свертки.....	70
6.3.5. Открытое хеширование.....	71
6.3.6. Закрытое хеширование.....	74
Ключевые термины .....	79
Краткие итоги .....	80
Материалы для практики.....	80
Литература .....	81
<b>7. Алгоритмы поиска в тексте .....</b>	<b>82</b>
7.1. Основные понятия задач поиска в тексте.....	82
7.2. Прямой поиск .....	83
7.3. Алгоритм Кнута, Морриса и Пратта .....	84
7.4. Алгоритм Бойера и Мура .....	86
Ключевые термины .....	89
Краткие итоги .....	89
Материалы для практики.....	90
Литература .....	91
<b>8. Алгоритмы поиска на основе деревьев .....</b>	<b>92</b>
8.1. Основные понятия задач поиска на основе деревьев .....	92
8.2. Двоичные (бинарные) деревья.....	92
8.3. Двоичные упорядоченные деревья.....	93
8.4. Случайные деревья .....	97
8.5. Оптимальные деревья.....	98
8.6. Сбалансированные по высоте деревья.....	98
8.7. Деревья цифрового (поразрядного) поиска.....	105
Ключевые термины .....	105
Краткие итоги .....	106
Материалы для практики.....	107
Литература .....	107

<b>9. Алгоритмы сжатия данных.....</b>	<b>109</b>
9.1. Основные понятия и методы сжатия данных.....	109
9.2. Метод Хаффмана.....	111
9.3. Кодовые деревья.....	115
Ключевые термины .....	120
Краткие итоги .....	121
Материалы для практики.....	122
Литература .....	123
<b>10. Алгоритмы сортировки массивов. Внутренняя сортировка .....</b>	<b>124</b>
10.1. Основные понятия сортировки .....	124
10.2. Оценка алгоритмов сортировки.....	125
10.3. Классификация алгоритмов сортировок.....	125
10.4. Бинарная пирамидальная сортировка .....	126
10.5. Сортировка методом Шелла .....	130
10.6. Быстрая сортировка Хоара .....	132
10.7. Сортировка слиянием .....	135
Ключевые термины .....	137
Краткие итоги .....	138
Материалы для практики.....	139
Литература .....	140
<b>11. Алгоритмы сортировки массивов. Внешняя сортировка .....</b>	<b>142</b>
11.1. Основные понятия алгоритмов внешних сортировок .....	142
11.2. Сортировка простым слиянием .....	144
11.3. Сортировка естественным слиянием .....	147
Ключевые термины .....	150
Краткие итоги .....	151
Материалы для практики.....	151
Литература .....	152
<b>12. Алгоритмы на графах. Алгоритмы обхода графа .....</b>	<b>153</b>
12.1. Основные понятия теории графов.....	153
12.2. Поиск в глубину .....	156
12.4. Поиск в ширину.....	157
Ключевые термины .....	158
Краткие итоги .....	160
Материалы для практики.....	160
Литература .....	161
<b>13. Алгоритмы на графах. Алгоритмы нахождения кратчайшего пути .....</b>	<b>162</b>
13.1. Алгоритмы поиска на графах.....	162
13.2. Алгоритм Дейкстры .....	162
13.3. Алгоритм Флойда.....	165
13.4. Переборные алгоритмы .....	167

13.4.1. Перебор с возвратом.....	167
13.4.2. Волновой алгоритм.....	170
Ключевые термины .....	171
Краткие итоги .....	171
Материалы для практики.....	172
Литература .....	173
<b>14. Решение задач на использование алгоритмов обработки данных .....</b>	<b>174</b>
14.1. Этапы решения задач на обработку данных .....	174
14.2. Алгоритмы сортировки данных.....	175
14.3. Алгоритмы на графах .....	179
14.4. Алгоритмы сжатия данных .....	187
Ключевые термины .....	194
Краткие итоги .....	194
Материалы для практики.....	194
Литература .....	196
<b>15. Тестовые задания .....</b>	<b>197</b>
15.1. Тест по теме «Трудоёмкость алгоритмов и рекурсия» .....	197
Вариант 1 .....	197
Вариант 2 .....	199
Вариант 3 .....	201
15.2. Тест по теме «Алгоритмы поиска, хеширования и сжатия данных».....	203
Вариант 1 .....	203
Вариант 2 .....	205
Вариант 3 .....	208
15.3. Тест по теме «Алгоритмы сортировки массивов. Алгоритмы на графах».....	211
Вариант 1 .....	211
Вариант 2 .....	214
Вариант 3 .....	217

## Предисловие

Язык программирования С++ был разработан датским ученым Бьёрном Страуструпом в начале 80-х годов, первоначально как объектно-ориентированное расширение языка С. В настоящее время С++ является одним из наиболее мощных и широко распространенных языков программирования. Язык характеризуется своей универсальностью, он успешно применяется для решения разнообразных задач прикладного и системного программирования с использованием различных парадигм программирования – процедурной, объектно-ориентированной, модульной.

Настоящее пособие «Алгоритмы компьютерной обработки данных» обобщает и систематизирует различные абстракции данных, поддерживаемые в С++, в контексте применяемых для их обработки алгоритмов. Основным содержанием курса являются лекционно-практические тематические разделы, которые следуют в порядке, соответствующем последовательности изучения алгоритмов компьютерной обработки данных при подготовке студентов. В качестве программной реализации выбран язык С++ и использована среда MS Visual Studio 2010.

Представленный в курсе теоретический материал и набор упражнений покрывает основные синтаксические и семантические аспекты языка С++ в объеме вузовских программ по структурам и алгоритмам компьютерной обработки данных для специальности 351500 Математическое обеспечение и администрирование информационных систем и направлений подготовки 010500.62 Математическое обеспечение и администрирование информационных систем, 010300.62 Фундаментальная информатика и информационные технологии. Отдельные разделы могут быть использованы при обучении программированию студентов специальности 030100 Информатика и направлений подготовки 540200 Физико-математическое образование (профиль 540203 Информатика), 050100.62 Педагогическое образование (профиль Информатика).

Теоретический и практический материал опирается на сформированные ранее базовые знания обучающихся по основам алгоритмизации, на умения работать со структурированными типами данных языка С++ и владение базовыми алгоритмами обработки простых и структурированных данных. В пособии продолжается реализация идеи процедурной парадигмы программирования, так как обучение знанию алгоритмов обработки структурированных данных и построение на их основе решения различных классов задач способствует формированию определенного стиля мышления и культуры. Формируется база для перехода к изучению альтернативных парадигм современного программирования.

Курс представлен следующими тематическими разделами, включающими теоретический материал и задания для практической работы:

- трудоемкость алгоритмов;
- рекурсия и рекурсивные алгоритмы;

- алгоритмы поиска, хеширования и сжатия данных;
- алгоритмы сортировки массивов;
- алгоритмы на графах.

Каждая тема начинается с краткой аннотации и изложения теоретического материала, на основе которого построено объяснение рассматриваемого способа обработки структурированных данных, изложение сути методов и применяемых алгоритмов, технологических особенностей программирования. При необходимости в тексте приводится справочный материал. В материале широко представлены многочисленные примеры программных кодов с комментариями, в которых раскрываются алгоритмические подходы к решению задач. Для закрепления изученного материала и приобретения навыков программирования предусмотрена система практических заданий, которые можно использовать на аудиторных занятиях, а также для самостоятельной работы в соответствии с рассматриваемой тематикой.

Пособие написано на основе курса лекций и лабораторно-практических занятий по дисциплине «Структуры и алгоритмы компьютерной обработки данных» со студентами факультета математики, физики и информатики ТГПУ им. Л.Н. Толстого. Для базовой подготовки студентов, обучающихся на основе материалов пособия, необходимо знание основ программирования, умение работать с массивами, строками и реализовывать метод процедурной абстракции средствами языка C++.



# 1. Алгоритмы обработки данных

## Краткая аннотация

В данной теме рассматриваются понятие ресурсной эффективности алгоритмов посредством анализа асимптотических функций временной и емкостной сложности, приводится классификация алгоритмов на основе функции временной сложности, рассматриваются общие методы оценки трудоемкости алгоритмов.

## Цель изучения темы

Изучить понятие и классификацию алгоритмов обработки данных, трудоемкость алгоритмов и методов ее оценки, научиться выработке критериев и оценке трудоемкости алгоритмов с учетом критериев на примере реализаций и задач на языке C++.

### 1.1. Понятие «алгоритм обработки данных»

Использование вычислительной техники при решении задач в течение многих десятилетий позволило выстроить общую схему подхода к работе над сформулированной проблемой. Решение поставленных задач укладывается в так называемые этапы решения задач, которые начинаются с информационной модели (работа над условием) и завершаются построением компьютерной модели (реализация алгоритма средствами языков программирования).

Понятие «*алгоритм обработки данных*» в компьютерных науках используется для описания метода решения задачи, который в дальнейшем возможно реализовать в выбранной среде программирования. Тщательная разработка алгоритма является весьма эффективной частью процесса решения задачи в любой области применения. При разработке алгоритма для реальной задачи значительные усилия должны быть потрачены на осознание степени ее сложности, выяснение ограничений на входные данные, разбиение задачи на менее трудоемкие подзадачи.

Алгоритм не должен быть привязан к конкретной реализации. В силу разнообразия используемых средств программирования, их требований к аппаратным ресурсам и платформенной зависимости сходные по структуре, но различные в реализации, алгоритмы могут выдавать отличающиеся по эффективности результаты. При этом некоторые среды программирования содержат встроенные библиотечные функции, реализующие базовые алгоритмы обработки данных (например, в MS Visual Studio 2010 в библиотеки C++ входит функция быстрой сортировки массивов данных). Чтобы решения были переносимыми и оставались актуальными, не рекомендуется их ориентировать на процедурную реализацию среды. Поэтому главным в рассматриваемом подходе является выбор метода решения с учетом специфики задачи. Адаптация к среде осуществляется позднее.

Выбор того или иного метода обработки данных определяется не только сложностью задачи. Учитывать необходимо и массовость применения разработанного кода: при однократном или редком обращении к реализации предпочтительнее бывают простые алгоритмы, которые несложны в разработке. При этом, однако, допускается возможным увеличение времени работы программы.

Массовое использование алгоритмов обработки данных требует поиска наилучшего алгоритма решения. Такой процесс бывает весьма сложен, так как требует выработки критериев оценки и применения математических методов для получения количественных характеристик. Направление компьютерных наук, занимающееся изучением оценки эффективности алгоритмов, называется *анализом алгоритмов*.

## 1.2. Ресурсная эффективность алгоритмов

Определение ресурсной эффективности алгоритмов – необходимая составляющая этапа анализа разработанного программного обеспечения. Повышение ресурсной эффективности вычислительных алгоритмов актуально при обработке больших объемов данных, когда аппаратных и/или программных ресурсов может быть недостаточно для корректного завершения работы программного кода.

Наиболее значимыми характеристиками ресурсной эффективности алгоритмов являются оценки временной и емкостной сложности, отражающие ресурсы процессора, оперативной памяти, а также внешних носителей данных (при использовании).

Под *трудоемкостью алгоритма*  $A$  на входе  $D$  будем понимать количество элементарных операций, которые учитываются при анализе алгоритма. Под *худшим случаем* трудоемкости понимают наибольшее количество операций, задаваемых алгоритмом  $A$  на всех входах  $D$  определенной размерности  $n$ . Определим *лучший случай* трудоемкости как наименьшее количество операций в аналогичном алгоритме и при той же размерности входа. *Средний случай* трудоемкости определяется средним количеством операций рассматриваемого алгоритма и входных данных. Зависимость трудоемкости алгоритма  $A$  от значения параметров на входе  $D$  определяет *функцию трудоемкости алгоритма*  $A$  для входа  $D$ .

Классический анализ алгоритмов в данном контексте связан, прежде всего, с оценкой временной сложности. Большинство алгоритмов имеют основной параметр, который в значительной степени влияет на время выполнения операций. Если же определяющих параметров несколько, то, как правило, один из них выражается как функция от остальных. Иногда используют и такой подход: рассматривают только один параметр, считая остальные константами.

Результатом анализа является *асимптотическая оценка* выполняемых алгоритмом операций в зависимости от длины входа, которая указы-

вает порядок роста функции и результаты сравнения работы алгоритмов для больших данных. При этом оценка на реальных данных отличается от асимптотической тем, что она ориентирована на конкретные длины входов и число выполняемых алгоритмом операций.

*Временная сложность алгоритма* определяется асимптотической оценкой функции трудоемкости алгоритма для худшего случая, обозначается  $O(f(n))$  и читается как «О большое» или «О-нотация». Асимптотический класс функций  $O$  включает в себя как средний, так и лучший случай, потому что запись  $O(f(n))$  обозначает класс функций, скорость роста которых не более, чем  $f(n)$  с точностью до некоторой положительной константы. В зависимости от вида функции  $f(n)$  выделяют следующие классы сложности алгоритмов.

***Классы сложности алгоритмов  
в зависимости от функции трудоемкости***

<b><i>Вид <math>f(n)</math></i></b>	<b><i>Характеристика класса алгоритмов</i></b>
<b>1</b>	Большинство инструкций большинства функций запускается один или несколько раз. Если все инструкции программы обладают таким свойством, то время выполнения программы <i>постоянно</i> .
<b><math>\log N</math></b>	Когда время выполнения программы является <i>логарифмическим</i> , программа становится медленнее с ростом $N$ . Такое время выполнения обычно присуще программам, которые сводят большую задачу к набору меньших подзадач, уменьшая на каждом шаге размер задачи на некоторый постоянный фактор. Будем рассматривать время выполнения, являющееся небольшой по величине константой. Изменение основания не сильно сказывается на изменении значения логарифма: при $N=1\,000$ , $\log N = 3$ , если основание равно 10, или порядка 10, если основание равно 2; когда $N=1\,000\,000$ , значения $\log N$ увеличивается в два раза. При удвоении значения параметра $\log N$ растет на постоянную величину, а удваивается лишь тогда, когда $N$ достигает $N^2$ .
<b><math>N</math></b>	Когда время выполнения программы является <i>линейным</i> , это обычно значит, что каждый входной элемент подвергается небольшой обработке. Когда $N$ равно миллиону, таким же и является время выполнения. Когда $N$ удваивается, то же происходит и со временем выполнения. Эта ситуация оптимальна для алгоритма, который должен обработать $N$ вводов (или произвести $N$ выводов).

$N \log N$	Время выполнения, пропорциональное $N \log N$ , возникает тогда, когда алгоритм решает задачу, разбивая ее на меньшие подзадачи, решая их независимо и затем объединяя решения. Время выполнения такого алгоритма равно $N \log N$ . Когда $N=1\,000\,000$ , $N \log N \approx 20\,000\,000$ . Когда $N$ удваивается, тогда время выполнения более чем удваивается.
$N^2$	Когда время выполнения алгоритма является <i>квадратичным</i> , он полезен для практического использования при решении относительно небольших задач. Квадратичное время выполнения обычно появляется в алгоритмах, которые обрабатывают все пары элементов данных (возможно, в цикле двойного уровня вложенности). Когда $N=1\,000$ , время выполнения равно одному миллиону. Когда $N$ удваивается, время выполнения увеличивается вчетверо.
$N^3$	Похожий алгоритм, который обрабатывает тройки элементов данных (возможно, в цикле тройного уровня вложенности), имеет <i>кубическое</i> время выполнения и практически применим лишь для малых задач. Когда $N=100$ , время выполнения равно одному миллиону. Когда $N$ удваивается, время выполнения увеличивается в восемь раз.
$2^N$	Лишь несколько алгоритмов с <i>экспоненциальным</i> временем выполнения имеет практическое применение, хотя такие алгоритмы возникают естественным образом при попытках прямого решения задачи, например полного перебора. Когда $N=20$ , время выполнения имеет порядок одного миллиона. Когда $N$ удваивается, время выполнения увеличивается экспоненциально.

На основании математических методов исследования асимптотических функций трудоемкости на бесконечности выделены пять классов алгоритмов.

*Класс  $\pi 0$*  – это класс быстрых алгоритмов с постоянным временем выполнения, их функция трудоемкости  $O(1)$ . Промежуточное состояние занимают алгоритмы со сложностью  $O(\log N)$ , которые также относят к данному классу.

*Класс  $\pi P$*  – это класс рациональных или полиномиальных алгоритмов, функция трудоемкости которых определяется полиномиально от входных параметров. Например,  $O(N)$ ,  $O(N^2)$ ,  $O(N^3)$ .

*Класс  $\pi L$*  – это класс субэкспоненциальных алгоритмов со степенью трудоемкости  $O(N \log N)$ .

*Класс  $\pi E$*  – это класс собственно экспоненциальных алгоритмов со степенью трудоемкости  $O(2^N)$ .