

# Matters Computational

[DOWNLOAD HERE](#)

1;Preface;5 2;Contents;7 3;Part I: Low level algorithms ;15 3.1;Chapter 1:Bit wizardry ;16 3.1.1;Trivia;16 3.1.2;Operations on individual bits;21 3.1.3;Operations on low bits or blocks of a word;22 3.1.4;Extraction of ones, zeros, or blocks near transitions;25 3.1.5;Computing the index of a single set bit;27 3.1.6;Operations on high bits or blocks of a word;28 3.1.7;Functions related to the base-2 logarithm;31 3.1.8;Counting the bits and blocks of a word;32 3.1.9;Words as bitsets;37 3.1.10;Index of the i-th set bit;39 3.1.11;Avoiding branches;39 3.1.12;Bit-wise rotation of a word;41 3.1.13;Binary necklaces ;43 3.1.14;Reversing the bits of a word;47 3.1.15;Bit-wise zip;52 3.1.16;Gray code and parity;55 3.1.17;Bit sequency ;60 3.1.18;Powers of the Gray code ;62 3.1.19;Invertible transforms on words ;63 3.1.20;Scanning for zero bytes;69 3.1.21;Inverse and square root modulo  $2^n$ ;70 3.1.22;Radix -2 (minus two) representation;72 3.1.23;A sparse signed binary representation;75 3.1.24;Generating bit combinations;76 3.1.25;Generating bit subsets of a given word;82 3.1.26;Binary words in lexicographic order for subsets;84 3.1.27;Fibonacci words ;88 3.1.28;Binary words and parentheses strings ;92 3.1.29;Permutations via primitives ;94 3.1.30;CPU instructions often missed;96 3.1.31;Some space filling curves ;97 3.2;Chapter 2:Permutations and their operations ;116 3.2.1;Basic definitions and operations;116 3.2.2;Representation as disjoint cycles;118 3.2.3;Compositions of permutations;119 3.2.4;In-place methods to apply permutations to data;123 3.2.5;Random permutations;125 3.2.6;The revbin permutation;132 3.2.7;The radix permutation;135 3.2.8;In-place matrix transposition;136 3.2.9;Rotation by triple reversal;137 3.2.10;The zip permutation;139 3.2.11;The XOR permutation;141 3.2.12;The Gray permutation;142 3.2.13;The reversed Gray permutation;145 3.3;Chapter 3:Sorting and searching ;148 3.3.1;Sorting algorithms;148 3.3.2;Binary search;155 3.3.3;Variants of sorting methods;156 3.3.4;Searching in unsorted arrays;161 3.3.5;Determination of equivalence classes;162 3.4;Chapter 4:Data structures ;167 3.4.1;Stack (LIFO);167 3.4.2;Ring buffer;169 3.4.3;Queue (FIFO);170 3.4.4;Deque (double-ended queue);172 3.4.5;Heap and priority queue;174 3.4.6;Bit-array;178 3.4.7;Left-right array;180 4;Part II:Combinatorial generation ;185 4.1;Chapter 5: Conventions and considerations;186 4.1.1;Representations and orders;186 4.1.2;Ranking, unranking, and counting;186

4.1.3;Characteristics of the algorithms;187 4.1.4;Optimization techniques;188 4.1.5;Implementations, demo-programs, and timings;188 4.2;Chapter 6:Combinations ;190 4.2.1;Binomial coefficients;190 4.2.2;Lexicographic and co-lexicographic order;191 4.2.3;Order by prefix shifts (cool-lex);194 4.2.4;Minimal-change order;196 4.2.5;The Eades-McKay strong minimal-change order;197 4.2.6;Two-close orderings via endo/enup moves;200 4.2.7;Recursive generation of certain orderings;205 4.3;Chapter 7 :Compositions ;208 4.3.1;Co-lexicographic order;208 4.3.2;Co-lexicographic order for compositions into exactly k parts;210 4.3.3;Compositions and combinations;212 4.3.4;Minimal-change orders;213 4.4;Chapter 8:Subsets ;216 4.4.1;Lexicographic order;216 4.4.2;Minimal-change order;218 4.4.3;Ordering with De Bruijn sequences;222 4.4.4;Shifts-order for subsets;222 4.4.5;k-subsets where k lies in a given range;224 4.5;Chapter 9:Mixed radix numbers ;231 4.5.1;Counting (lexicographic) order;231 4.5.2;Minimal-change (Gray code) order;234 4.5.3;gslex order;238 4.5.4;endo order;240 4.5.5;Gray code for endo order;242 4.5.6;Fixed sum of digits;243 4.6;Chapter 10:Permutations ;246 4.6.1;Factorial representations of permutations;246 4.6.2;Lexicographic order;256 4.6.3;Co-lexicographic order;257 4.6.4;An order from reversing prefixes;259 4.6.5;Minimal-change order (Heap's algorithm);262 4.6.6;Lipski's Minimal-change orders; EAN/ISBN : 9783642147647 Publisher(s): Springer, Berlin  
Discussed keywords: Algorithmen Format: ePub/PDF Author(s): Arndt, Jrg

[DOWNLOAD HERE](#)

Similar manuals:

[Matters Computational](#)