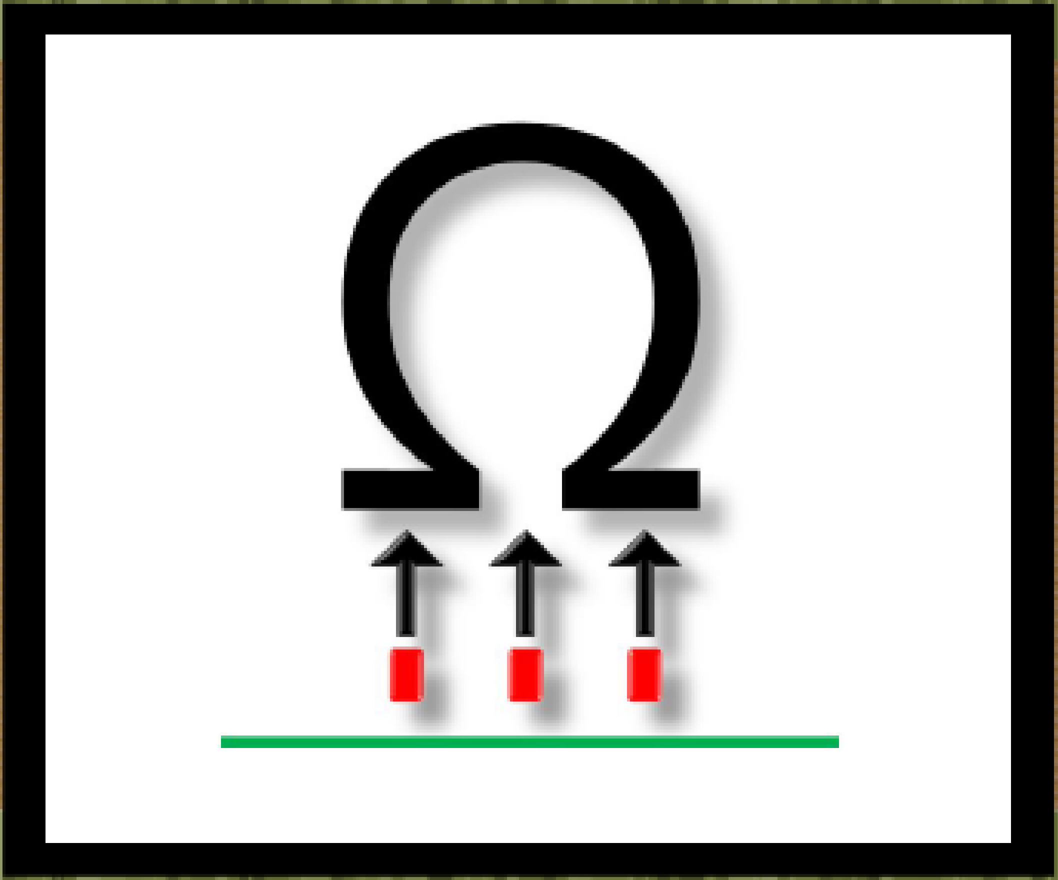
Competitive Programming 3

The New Lower Bound of Programming Contests.



Steven Halim
Felix Halim

HANDBOOK FOR ACM ICPC AND IOI CONTESTANTS
2013

Contents

Foreword

A long time ago (on the 11th of November in 2003, Tuesday, 3:55:57 UTC), I received an e-mail with the following message:

> "I should say in a simple word that with the UVa Site, you have given birth to a new CIVILIZATION and with the books you write (he meant "Programming Challenges: The Programming Contest Training Manual" [60], coauthored with Steven Skiena), you inspire the soldiers to carry on marching. May you live long to serve the humanity by producing super-human programmers."

Although that was clearly an exaggeration, it did cause me to think. I had a dream: to create a community around the project I had started as a part of my teaching job at UVa, with people from all around the world working together towards the same ideal. With a little searching, I quickly found a whole online community running a web-ring of sites with excellent tools that cover and provide whatever the UVa site lacked.

To me, 'Methods to Solve' by Steven Halim, a very young student from Indonesia, was one of the more impressive websites. I was inspired to believe that the dream would become real one day, because in this website lay the result of the hard work of a genius of algorithms and informatics. Moreover, his declared objectives matched the core of my dream: to serve humanity. Even better, he has a brother with similar interests and capabilities, Felix Halim.

It's a pity that it takes so much time to start a real collaboration, but life is like that. Fortunately, all of us have continued working together in a parallel fashion towards the realization of that dream—the book that you have in your hands now is proof of that.

I can't imagine a better complement for the UVa Online Judge. This book uses lots of examples from UVa carefully selected and categorized both by problem type and solving technique, providing incredibly useful help for the users of the site. By mastering and practicing most programming exercises in this book, a reader can easily solve at least 500 problems in the UVa Online Judge, which will place them in the top 400-500 amongst $\approx$100000 UVa OJ users.

It's clear that the book "Competitive Programming: Increasing the Lower Bound of Programming Contests" is suitable for programmers who want to improve their ranks in upcoming ICPC regionals and IOIs. The two authors have gone through these contests (ICPC and IOI) themselves as contestants and now as coaches. But it's also an essential colleague for newcomers—as Steven and Felix say in the introduction 'the book is not meant to be read once, but several times'.

Moreover, it contains practical C++ source code to implement given algorithms. Understanding a problem is one thing, but knowing the algorithm to solve it is another, and implementing the solution well in short and efficient code is tricky. After you have read this extraordinary book three times you will realize that you are a much better programmer and, more importantly, a happier person.

@ ACM ICPC World Finals Warsaw 2012
L-R: Fredrik Niemelä, Carlos, Miguel Revilla, Miguel Jr, Felix, Steven
UVa
Online Judge
IBM

Miguel A. Revilla, University of Valladolid
UVa Online Judge site creator;
ACM-ICPC International Steering Committee Member and Problem Archivist
`http://uva.onlinejudge.org; http://livearchive.onlinejudge.org`

Preface

This book is a must have for every competitive programmer. Mastering the contents of this book is a necessary (but maybe not sufficient) condition if one wishes to take a leap forward from being just another ordinary coder to being among one of the world's finest programmers.

Typical readers of this book would include:

1. University students who are competing in the annual ACM International Collegiate Programming Contest (ICPC) [66] Regional Contests (including the World Finals),
2. Secondary or High School Students who are competing in the annual International Olympiad in Informatics (IOI) [34] (including the National or Provincial Olympiads),
3. Coaches who are looking for comprehensive training materials for their students [24],
4. Anyone who loves solving problems through computer programs. There are numerous programming contests for those who are no longer eligible for ICPC, including TopCoder Open, Google CodeJam, Internet Problem Solving Contest (IPSC), etc.

Prerequisites

This book is *not* written for novice programmers. This book is aimed at readers who have at least basic knowledge in programming methodology, are familiar with at least one of these programming languages (C/C++ or Java, preferably both), have passed a basic data structures and algorithms course (typically taught in year one of Computer Science university curricula), and understand simple algorithmic analysis (at least the big-O notation). In the third edition, more content has been added so that this book can also be used as a *supplementary reading* for a basic *Data Structures and Algorithms* course.

To ACM ICPC Contestants

2012 World Finals
acm International Collegiate Programming Contest
IBM event sponsor
ICPC 2012
University of Warsaw
HOST
Zi Chun Koh, Trinh Tuan Phuong, Harta Wijaya
Steven Halim

We know that one cannot probably win the ACM ICPC regional just by mastering the contents of the *current version (third edition)* of this book. While we have included a lot of materials in this book—much more than in the first two editions—we are aware that much more than what this book can offer is required to achieve that feat. Some additional pointers to useful references are listed in the chapter notes for readers who are hungry for more. We believe, however, that your team will fare much better in future ICPCs after mastering the contents of this book. We hope that this book will serve as both inspiration and motivation for your 3-4 year journey competing in ACM ICPCs during your University days.

To IOI Contestants

L-R: Daniel, Mr Cheong, Raymond, Steven, Zhan Xiong, Dr Roland, Chuanqi
2010
IOI
Waterloo
Ontario Canada
EXIT

Much of our advice for ACM ICPC contestants applies to you too. The ACM ICPC and IOI syllabi are largely similar, except that IOI, *for now*, currently excludes the topics listed in the following Table 1. You can skip these items until your university years (when you join that university's ACM ICPC teams). However, learning these techniques in advance may definitely be beneficial as some tasks in IOI can become easier with additional knowledge.

We know that one cannot win a medal in IOI just by mastering the contents of the *current version (third edition)* of this book. While we believe that many parts of the IOI syllabus has been included in this book—hopefully enabling you to achieve a respectable score in future IOIs—we are well aware that modern IOI tasks require keen problem solving skills and tremendous creativity—virtues that we cannot possibly impart through this static textbook. This book can provide knowledge, but the hard work must ultimately be done by you. With practice comes experience, and with experience comes skill. So, keep practicing!

Topic	**In This Book**
Data Structures: Union-Find Disjoint Sets	Section 2.4.2
Graph: Finding SCCs, Network Flow, Bipartite Graphs	Section 4.2.1, 4.6.3, 4.7.4
Math: BigInteger, Probability Theory, Nim Games	Section 5.3, 5.6, 5.8
String Processing: Suffix Trees/Arrays	Section 6.6
More Advanced Topics: A*/IDA*	Section 8.2
Many of the Rare Topics	Chapter 9

Table 1: Not in IOI Syllabus [20] *Yet*

To Teachers and Coaches

This book is used in Steven's CS3233 - 'Competitive Programming' course in the School of Computing at the National University of Singapore. CS3233 is conducted in 13 teaching weeks using the following lesson plan (see Table 2). The PDF slides (only the public version) are given in the companion web site of this book. Fellow teachers/coaches should feel free to modify the lesson plan to suit students' needs. Hints or brief solutions of the **non-starred** written exercises in this book are given at the back of each chapter. Some of the **starred** written exercises are quite challenging and have neither hints nor solutions. These can probably be used as exam questions or contest problems (of course, solve them first!).

This book is also used as a supplementary reading in Steven's CS2010 - 'Data Structures and Algorithms' course, mostly for the implementation of several algorithms and written/programming exercises.

Wk	Topic	In This Book
01	Introduction	Ch 1, Sec 2.2, 5.2, 6.2-6.3, 7.2
02	Data Structures & Libraries	Chapter 2
03	Complete Search, Divide & Conquer, Greedy	Section 3.2-3.4; 8.2
04	Dynamic Programming 1 (Basic ideas)	Section 3.5; 4.7.1
05	Dynamic Programming 2 (More techniques)	Section 5.4; 5.6; 6.5; 8.3
06	Mid-Semester Team Contest	Chapter 1 - 4; parts of Ch 9
-	Mid-Semester Break	(homework)
07	Graph 1 (Network Flow)	Section 4.6; parts of Ch 9
08	Graph 2 (Matching)	Section 4.7.4; parts of Ch 9
09	Mathematics (Overview)	Chapter 5
10	String Processing (Basic skills, Suffix Array)	Chapter 6
11	(Computational) Geometry (Libraries)	Chapter 7
12	More Advanced Topics	Section 8.4; parts of Ch 9
13	Final Team Contest	Chapter 1-9 and maybe more
-	No final exam	-

Table 2: Lesson Plan of Steven's CS3233

For *Data Structures and Algorithms* Courses

The contents of this book have been expanded in this edition so that the *first four* chapters of this book are more accessible to *first year* Computer Science students. Topics and exercises that we have found to be relatively difficult and thus unnecessarily discouraging for first timers have been moved to the now bulkier Chapter 8 or to the new Chapter 9. This way, students who are new to Computer Science will perhaps not feel overly intimidated when they peruse the first four chapters.

Chapter 2 has received a major update. Previously, Section 2.2 was just a casual list of classical data structures and their libraries. This time, we have expanded the write-up and added lots of written exercises so that this book can also be used to support a *Data Structures* course, especially in the terms of *implementation* details.

The four problem solving paradigms discussed in Chapter 3 appear frequently in typical *Algorithms* courses. The text in this chapter has been expanded and edited to help new Computer Science students.

Parts of Chapter 4 can also be used as a supplementary reading or *implementation* guide to enhance a *Discrete Mathematics* [57, 15] or a basic *Algorithms* course. We have also provide some new insights on viewing Dynamic Programming techniques as algorithms on DAGs. Such discussion is currently still regrettably uncommon in many Computer Science textbooks.

To All Readers

Due to its diversity of coverage and depth of discussion, this book is *not* meant to be read once, but several times. There are many written (≈ 238) and programming exercises (≈ 1675) listed and spread across almost every section. You can skip these exercises at first if the solution is too difficult or requires further knowledge and technique, and revisit them after studying other chapters of this book. Solving these exercises will strengthen your understanding of the concepts taught in this book as they usually involve interesting applications, twists or variants of the topic being discussed. Make an effort to attempt them—time spent solving these problems will definitely not be wasted.

We believe that this book is and will be relevant to many university and high school students. Programming competitions such as the ICPC and IOI are here to stay, at least for many years ahead. New students should aim to understand and internalize the basic knowledge presented in this book before hunting for further challenges. However, the term 'basic' might be slightly misleading—please check the table of contents to understand what we mean by 'basic'.

As the title of this book may imply, the purpose of this book is clear: We aim to improve everyone's programming abilities and thus increase the *lower bound* of programming competitions like the ICPC and IOI in the future. With more contestants mastering the contents of this book, we hope that the year 2010 (when the first edition of this book was published) will be a watershed marking an accelerated improvement in the standards of programming contests. We hope to help more teams solve more (≥ 2) problems in future ICPCs and help more contestants to achieve greater (≥ 200) scores in future IOIs. We also hope to see many ICPC and IOI coaches around the world (especially in South East Asia) adopt this book for the aid it provides in mastering topics that students cannot do without in competitive programming contests. If such a proliferation of the required 'lower-bound' knowledge for competitive programming is achieved, then this book's primary objective of advancing the level of human knowledge will have been fulfilled, and we, as the authors of this book, will be very happy indeed.

Convention

There are lots of C/C++ code and also some Java code (especially in Section 5.3) included in this book. If they appear, they will be typeset in `this monospace font`.

For the C/C++ code in this book, we have adopted the frequent use of `typedef`s and macros—features that are commonly used by competitive programmers for convenience, brevity, and coding speed. However, we cannot use similar techniques for Java as it does not contain similar or analogous features. Here are some examples of our C/C++ code shortcuts:

```
// Suppress some compilation warning messages (only for VC++ users)
#define _CRT_SECURE_NO_DEPRECATE
```

```
// Shortcuts for "common" data types in contests
typedef long long        ll;        // comments that are mixed in with code
typedef pair<int, int>   ii;          // are aligned to the right like this
typedef vector<ii>       vii;
typedef vector<int>      vi;
#define INF 1000000000     // 1 billion, safer than 2B for Floyd Warshall's

// Common memset settings
//memset(memo, -1, sizeof memo); // initialize DP memoization table with -1
//memset(arr, 0, sizeof arr);                 // to clear array of integers

// We have abandoned the use of "REP" and "TRvii" since the second edition
// in order to reduce the confusion encountered by new programmers
```

The following shortcuts are frequently used in both our C/C++ and Java code:

```
// ans = a ? b : c;           // to simplify: if (a) ans = b; else ans = c;
// ans += val;            // to simplify: ans = ans + val; and its variants
// index = (index + 1) % n;          // index++; if (index >= n) index = 0;
// index = (index + n - 1) % n;   // index--; if (index < 0) index = n - 1;
// int ans = (int)((double)d + 0.5);     // for rounding to nearest integer
// ans = min(ans, new_computation);                      // min/max shortcut
// alternative form but not used in this book: ans <?= new_computation;
// some code use short circuit && (AND) and || (OR)
```

Problem Categorization

As of 24 May 2013, Steven and Felix—combined—have solved 1903 UVa problems ($\approx 46.45\%$ of the entire UVa problemset). About ≈ 1675 of them are discussed and categorized in this book. Since late 2011, some Live Archive problems have also been integrated in the UVa Online Judge. In this book, we use *both* problem numberings, but the primary sort key used in the index section of this book is the UVa problem number.

These problems are categorized according to a *'load balancing'* scheme: If a problem can be classified into two or more categories, it will be placed in the category with a lower number of problems. This way, you may find that some problems have been 'wrongly' categorized, where the category that it appears in might not match the technique that you have used to solve it. We can only guarantee that if you see problem X in category Y, then you know that *we* have managed to solve problem X with the technique mentioned in the section that discusses category Y.

We have also limited each category to at most 25 (TWENTY FIVE) problems, splitting them into separate categories when needed.

If you need hints for any of the problems (that we have solved), flip to the handy index at the back of this book instead of flipping through each chapter—it might save you some time. The index contains a list of UVa/LA problems, ordered by their problem number (do a binary search!) and augmented by the pages that contain discussion of said problems (and the data structures and/or algorithms required to solve that problem). In the third edition, we allow the hints to span more than one line so that they can be more meaningful.

Utilize this categorization feature for your training! Solving at least a few problems from each category (especially the ones we have highlighted as **must try ***) is a great way to diversify your problem solving skillset. For conciseness, we have limited ourselves to a maximum of 3 highlights per category.

Changes for the Second Edition

There are *substantial* changes between the first and the second edition of this book. As the authors, we have learned a number of new things and solved hundreds of programming problems during the one year gap between these two editions. We also have received feedback from readers, especially from Steven's CS3233 class Sem 2 AY2010/2011 students, and have incorporated these suggestions in the second edition.

Here is a summary of the important changes for the second edition:

- The first noticeable change is the layout. We now have a greater information density on each page. The 2nd edition uses single line spacing instead of the 1.5 line spacing used in the 1st edition. The positioning of small figures is also enhanced so that we have a more compact layout. This is to avoid increasing the number of pages by too much while still adding more content.

- Some minor bugs in our code examples (both the ones displayed in the book and the soft copies provided in the companion web site) have been fixed. All code samples now have much more meaningful comments to aid in comprehension.

- Several language-related issues (typographical, grammatical or stylistic) have been corrected.

- Besides enhancing the discussion of many data structures, algorithms, and programming problems, we have also added these *new* materials in each chapter:

    1. Many new Ad Hoc problems to kick start this book (Section 1.4).
    2. A lightweight set of Boolean (bit-manipulation) techniques (Section 2.2), Implicit Graphs (Section 2.4.1), and Fenwick Tree data structures (Section 2.4.4).
    3. More DP: A clearer explanation of bottom-up DP, the $O(n \log k)$ solution for the LIS problem, the 0-1 Knapsack/Subset Sum, and DP TSP (using the bitmask technique) (Section 3.5.2).
    4. A reorganization of the graph material into: Graph Traversal (both DFS and BFS), Minimum Spanning Tree, Shortest Paths (Single-Source and All-Pairs), Maximum Flow, and Special Graphs. New topics include Prim's MST algorithm, a discussion of DP as a traversal on implicit DAGs (Section 4.7.1), Eulerian Graphs (Section 4.7.3), and the Augmenting Path algorithm (Section 4.7.4).
    5. A reorganization of mathematical techniques (Chapter 5) into: Ad Hoc, Java BigInteger, Combinatorics, Number Theory, Probability Theory, Cycle-Finding, Game Theory (new), and Powers of a (Square) Matrix (new). Each topic has been rewritten for clarity.
    6. Basic string processing skills (Section 6.2), more string-related problems (Section 6.3), including string matching (Section 6.4), and an enhanced Suffix Tree/Array explanation (Section 6.6).
    7. More geometry libraries (Chapter 7), especially on points, lines and polygons.
    8. A new Chapter 8, which contains discussion on problem decomposition, advanced search techniques (A*, Depth Limited Search, Iterative Deepening, IDA*), advanced DP techniques (more bitmask techniques, the Chinese Postman Problem, a compilation of common DP states, a discussion of better DP states, and some harder DP problems).

- Many existing figures in this book have been redrawn and enhanced. Many new figures have been added to help explain the concepts more clearly.
- The first edition is mainly written using from the viewpoint of the ICPC contestant and C++ programmer. The second edition is written to be more balanced and includes the IOI perspective. Java support is also strongly enhanced in the second edition. However, we do not support any other programming languages as of yet.
- Steven's 'Methods to Solve' website has now been fully integrated in this book in the form of 'one liner hints' for each problem and the useful problem index at the back of this book. Now, reaching 1000 problems solved in UVa online judge is no longer a wild dream (we believe that this feat is doable by a *serious* 4-year CS university undergraduate).
- Some examples in the first edition use old programming problems. In the second edition, these examples have been replaced/added with newer examples.
- $\approx$ 600 more programming exercises from the UVa Online Judge and Live Archive have been solved by Steven & Felix and added to this book. We have also added many more written exercises throughout the book with hints/short solutions as appendices.
- Short profiles of data structure/algorithm inventors have been adapted from Wikipedia [71] or other sources for this book. It is nice to know a little bit more about these inventors.

Changes for the Third Edition

We gave ourselves two years (skipping 2012) to prepare a *substantial* number of improvements and additional materials for the third edition of this book. Here is the summary of the important changes for the third edition:

- The third edition now uses a slightly larger font size (12 pt) compared to second edition (11 pt), a 9 percent increase. Hopefully many readers will find the text more readable this time. We also use larger figures. These decisions, however, have increased the number of pages and rendered the book thicker. We have also adjusted the left/right margin in odd/even pages to increase readability.
- The layout has been changed to start almost every section on a new page. This is to make the layout far easier to manage.
- We have added *many more* written exercises throughout the book and classifed them into **non-starred** (for self-checking purposes; hints/solutions are at the back of each chapter) and **starred *** versions (for extra challenges; no solution is provided). The written exercises have been placed close to the relevant discussion in the body text.
- $\approx$ 477 more programming exercises from the UVa Online Judge and Live Archive have been solved by Steven & Felix and consequently added to this book. We thus have maintained a sizeable $\approx$ 50% (to be precise, $\approx$ 46.45%) coverage of UVa Online Judge problems even as the judge has grown in the same period of time. These newer problems have been listed in an *italic font*. Some of the newer problems have replaced older ones as the **must try** problems. All programming exercises are now always placed at the end of a section.

- We now have proof that *capable* CS students can achieve $\geq$ 500 AC problems (from 0) in the UVa Online Judge in just one University semester (4 months) with this book.
- The *new* (or revised) materials, chapter by chapter:

  1. Chapter 1 contains a gentler introduction for readers who are new to competitive programming. We have elaborated on stricter Input/Output (I/O) formats in typical programming problems and common routines for dealing with them.
  2. We add one more linear data structure: 'deque' in Section 2.2. Chapter 2 now contains a more detailed discussion of almost all data structures discussed in this chapter, especially Section 2.3 and 2.4.
  3. In Chapter 3, we have a more detailed discussions of various Complete Search techniques: Nested loops, generating subsets/permutations iteratively, and recursive backtracking. New: An interesting trick to write and print Top-Down DP solutions, Discussion of Kadane's algorithm for Max 1D Range Sum.
  4. In Chapter 4, we have revised white/gray/black labels (legacy from [7]) to their standard nomenclature, renaming 'max flow' to 'network flow' in the process. We have also referred to the algorithm author's actual scientific paper for a better understanding of the original ideas of the algorithm. We now have new diagrams of the implicit DAG in classical DP problems found in Section 3.5.
  5. Chapter 5: We have included greater coverage of Ad Hoc mathematics problems, a discussion of an interesting Java BigInteger operation: `isProbablePrime`, added/expanded several commonly used Combinatorics formulae and modified sieve algorithms, expanded/revised sections on Probability Theory (Section 5.6), Cycle-finding (Section 5.7), and Game Theory (Section 5.8).
  6. Chapter 6: We rewrite Section 6.6 to have a better explanation of Suffix Trie/Tree/Array by reintroducing the concept of terminating character.
  7. Chapter 7: We trim this chapter into two core sections and improve the library code quality.
  8. Chapter 8: The harder topics that were listed in Chapter 1-7 in the 2^{nd} edition have now been relocated to Chapter 8 (or Chapter 9 below). New: Discussion of harder backtracking routine, State-Space search, meet in the middle, trick of using balanced BST as memo table, and a more comprehensive section about problem decomposition.
  9. New Chapter 9: Various rare topics that appear once a while in programming contests have been added. Some of them are easy, but many of them are hard and can be somewhat important score determinants in programming contests.

Supporting Websites

This book has an official companion web site at `sites.google.com/site/stevenhalim`, from which you can obtain a soft copy of sample source code and the (*public/simpler version*) of the) PDF slides used in Steven's CS3233 classes.

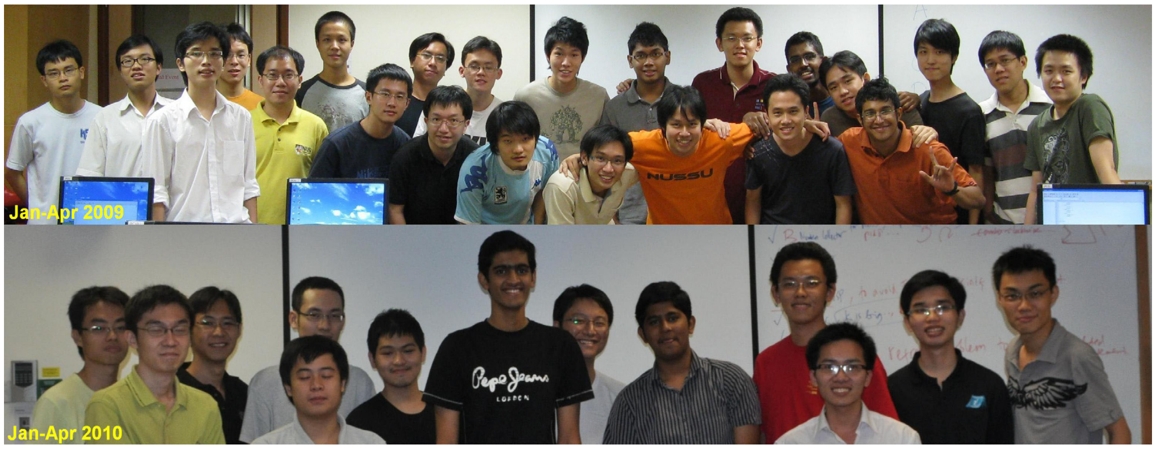
All programming exercises in this book are integrated in the `uhunt.felix-halim.net` tool and can be found in the UVa Online Judge at `uva.onlinejudge.org`

New in the third edition: Many algorithms now have interactive visualizations at:
`www.comp.nus.edu.sg/~stevenha/visualization`

Acknowledgments for the First Edition

From Steven: **I want to thank**

- God, Jesus Christ, and the Holy Spirit, for giving me talent and passion in competitive programming.
- my lovely wife, Grace Suryani, for allowing me to spend our precious time for this project.
- my younger brother and co-author, Felix Halim, for sharing many data structures, algorithms, and programming tricks to improve the writing of this book.
- my father Lin Tjie Fong and mother Tan Hoey Lan for raising us and encouraging us to do well in our study and work.
- the School of Computing, National University of Singapore, for employing me and allowing me to teach the CS3233 - 'Competitive Programming' module from which this book was born.
- NUS/ex-NUS professors/lecturers who have shaped my competitive programming and coaching skills: Prof Andrew Lim Leong Chye, Assoc Prof Tan Sun Teck, Aaron Tan Tuck Choy, Assoc Prof Sung Wing Kin, Ken, Dr Alan Cheng Holun.
- my friend Ilham Winata Kurnia for proof reading the manuscript of the first edition.
- fellow Teaching Assistants of CS3233 and ACM ICPC Trainers @ NUS: Su Zhan, Ngo Minh Duc, Melvin Zhang Zhiyong, Bramandia Ramadhana.
- my CS3233 students in Sem2 AY2008/2009 who inspired me to come up with the lecture notes and students in Sem2 AY2009/2010 who verified the content of the first edition of this book and gave the initial Live Archive contribution

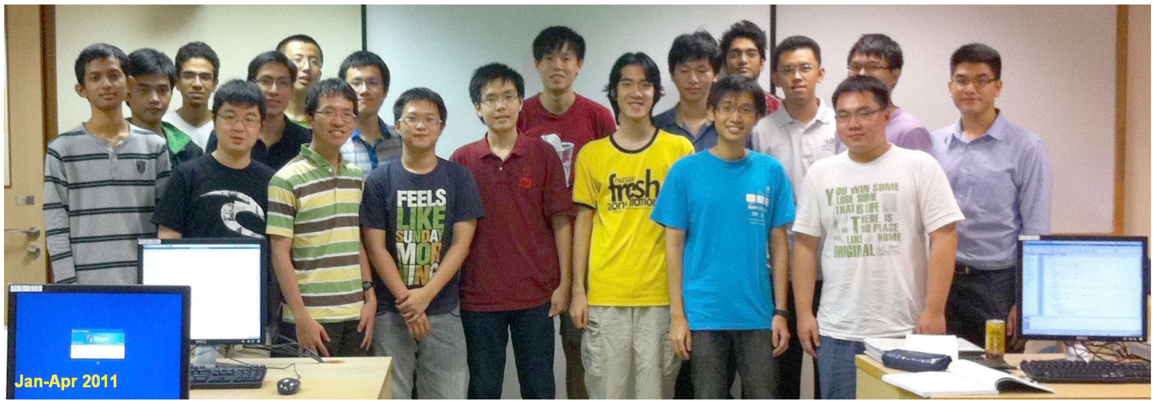


Acknowledgments for the Second Edition

From Steven: **Additionally, I also want to thank**

- the first ≈ 550 buyers of the 1st edition as of 1 August 2011 (this number is no longer updated). Your supportive responses encourage us!

- a fellow Teaching Assistant of CS3233 @ NUS: Victor Loh Bo Huai.
- my CS3233 students in Sem2 AY2010/2011 who contributed in both technical and presentation aspects of the second edition, in alphabetical order: Aldrian Obaja Muis, Bach Ngoc Thanh Cong, Chen Juncheng, Devendra Goyal, Fikril Bahri, Hassan Ali Askari, Harta Wijaya, Hong Dai Thanh, Koh Zi Chun, Lee Ying Cong, Peter Phandi, Raymond Hendy Susanto, Sim Wenlong Russell, Tan Hiang Tat, Tran Cong Hoang, Yuan Yuan, and one other student who prefers to be anonymous.



- the proof readers: Seven of CS3233 students above (underlined) plus Tay Wenbin.
- Last but not least, I want to re-thank my wife, Grace Suryani, for letting me do another round of tedious book editing process while she was pregnant with our first baby: Jane Angelina Halim.

Acknowledgments for the Third Edition

From Steven: **Again, I want to thank**

- the ≈ 2000 buyers of the 2nd edition as of 24 May 2013 (this number is no longer updated). Thanks :).