Your time machine is playing up
## Contents

**Introduction**  
Computational and Algorithmic Thinking  
Algorithmic Thinking and the Australian Curriculum  
Acknowledgments  

**Questions**  

**Applying Rules**  

**Logic**  

**Analysis**  
  - Searching  
  - Sorting  
  - Patterns  
  - Number of Routes  
  - How Many Ways?  
  - Analyse the Problem  

**Algorithms**  
  - Breadth-first Search  
  - Shortest Path  
  - Maximum Flow  
  - Single Pass Algorithms  
  - Dynamic Programming  
  - Two Person Games  
  - Greedy Algorithm  
  - Lists  
  - Working Backwards  
  - Chains  
  - Ad Hoc Algorithms
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solutions</td>
<td>81</td>
</tr>
<tr>
<td>Applying Rules</td>
<td>83</td>
</tr>
<tr>
<td>Logic</td>
<td>87</td>
</tr>
<tr>
<td>Analysis</td>
<td>93</td>
</tr>
<tr>
<td>Searching</td>
<td>93</td>
</tr>
<tr>
<td>Sorting</td>
<td>94</td>
</tr>
<tr>
<td>Patterns</td>
<td>96</td>
</tr>
<tr>
<td>Number of Routes</td>
<td>98</td>
</tr>
<tr>
<td>How Many Ways?</td>
<td>101</td>
</tr>
<tr>
<td>Analyse the Problem</td>
<td>105</td>
</tr>
<tr>
<td>Algorithms</td>
<td>109</td>
</tr>
<tr>
<td>Breadth-first Search</td>
<td>109</td>
</tr>
<tr>
<td>Shortest Path</td>
<td>121</td>
</tr>
<tr>
<td>Maximum Flow</td>
<td>125</td>
</tr>
<tr>
<td>Single Pass Algorithms</td>
<td>126</td>
</tr>
<tr>
<td>Dynamic Programming</td>
<td>128</td>
</tr>
<tr>
<td>Two Person Games</td>
<td>132</td>
</tr>
<tr>
<td>Greedy Algorithm</td>
<td>139</td>
</tr>
<tr>
<td>Lists</td>
<td>144</td>
</tr>
<tr>
<td>Working Backwards</td>
<td>146</td>
</tr>
<tr>
<td>Chains</td>
<td>148</td>
</tr>
<tr>
<td>Ad Hoc Algorithms</td>
<td>150</td>
</tr>
<tr>
<td>Development of Algorithms</td>
<td>161</td>
</tr>
<tr>
<td>Statistics</td>
<td>173</td>
</tr>
<tr>
<td>High Distinction Students</td>
<td>175</td>
</tr>
<tr>
<td>2011 Competition</td>
<td>175</td>
</tr>
<tr>
<td>2012 Competition</td>
<td>176</td>
</tr>
<tr>
<td>2013 Competition</td>
<td>177</td>
</tr>
<tr>
<td>2014 Competition</td>
<td>179</td>
</tr>
<tr>
<td>2015 Competition</td>
<td>181</td>
</tr>
</tbody>
</table>
Introduction

Computational and Algorithmic Thinking

The Computational and Algorithmic Thinking (CAT) competition, formerly known as the Australian Informatics Competition (AIC), is a pre-programming competition taken annually by more than 7000 school students from Australia, New Zealand and a number of other Asia-Pacific countries. It is run by the Australian Informatics Olympiad Committee (AIOC) and is intended to serve a number of purposes. Firstly, it introduces students and their teachers to algorithmic thinking, now a required component of the Australian Curriculum, through engaging problems without the need for programming skills. Secondly, it identifies students with the capacity to program and points them towards resources through which they can pursue this, and to further competitions which they can enter with a view to developing their programming skills. Finally, it is the first in a series of events that identify students who will represent Australia at the annual International Olympiad in Informatics (IOI). (The term informatics is used in Europe as a synonym for computer science.)

The CAT has traditionally been a pen and paper competition, but in 2014 a decision was made to introduce an online version of the competition from 2015. In addition, a further division of the competition was introduced for Upper Primary students (Years 5 and 6). The questions in the CAT are designed to be quick to solve and highly approachable, and range in difficulty from very easy to challenging. The contest employs a mixture of multiple-choice and integer answers, and incorporates unique ‘three-stage tasks’ that encourage students to develop informal algorithms and apply them to test data of increasing size.

The multiple-choice questions can be classified into four broad categories:

- **applying the rules** questions, that ask students to apply a well-defined set of rules to given data
- **logic** questions, that use non-algorithmic puzzles to encourage rigorous reasoning and case analysis
- **analysis** questions, where students are asked to understand the behaviour of an algorithm that solves a given problem
- **algorithm** questions, that encourage students to develop an informal algorithm to solve a given puzzle.

Explanations of these categories and their relevance are given in the ‘Solutions’ section of this book.
The integer answer questions are typically algorithmic. Students must devise and follow some repeated systematic procedure. For each question the students are given three sets of data of increasing size. These questions are explicitly designed to entice students into formulating algorithms. The first data set is often small enough to be solved by ad hoc techniques. By the second, students should have a feel for the problem and be developing systematic procedures for solving the problem. These procedures can then be applied to the third data set.

Algorithmic and analysis questions are often variations on well-known classes of problems such as shortest path problems (algorithm questions) and searching algorithms (analysis questions), and some students may have already encountered them. Other questions are not standard and students have to devise algorithms from the context of the problem.

The ‘Questions’ section of the book presents the questions, organised by category and problem type. The ‘Solutions’ section gives the solutions and answers. Each category is given a general introduction, describing its relevance to information technology in general and programming in particular. Each problem type also has an introduction, including the practical applications of the problem type and an outline of the method of solution.

Algorithmic Thinking and the Australian Curriculum

The Australian Curriculum has been developed over the last few years and is being adopted by all states and territories with implementation F – 10 scheduled to be complete by 2018. In the Technologies learning area of the Curriculum, there are two distinct but related subjects, one of which is Digital Technologies. For many teachers, the content of this subject will present quite a challenge, as it requires them to teach algorithmic thinking from the Foundation year and to introduce coding from as early as Year 3. Previously, this has been the preserve of a small number of courses in the senior years of the curriculum, but there is a worldwide demand for greater coding skills as a part of core education; Australia is not alone in promoting this type of thinking as a part of the compulsory curriculum. Victoria has taken a further initiative and included algorithmic thinking as a part of the mathematics curriculum. Other states are also looking at how this implementation will work in practice and who can best deliver the algorithmic content.

There is a rapidly growing variety of resources to teach various different programming languages, but there is little point in learning a programming language without a good understanding of the algorithmic thinking which sits behind any purposeful computer program. In addition to providing many questions which promote this type of thinking, this book contains a background chapter on what an algorithm is and explores how questions can be analysed from an algorithmic perspective. We would prefer that teachers did not think of algorithmic thinking as yet another thing which they have to teach, but rather as a pedagogical approach to problem solving in general, a skill which will be transferable across many disciplines.

Acknowledgments

The AIOC is a department of the Australian Mathematics Trust (AMT), which provides both financial and administrative support to the informatics program. Indeed, the formation of the AIOC in 1999 was an initiative of Professor Peter Taylor, then Executive Director of the AMT. Without the support of the AMT, the AIOC could not survive.

The Chair of the CAT problems committee since 2007 has been David Clark, who is also the author of this book. For David, this has been a labour of love. His relentless and
continuing enthusiasm is amazing. The problems in the CAT have been devised by past and present members of the CAT problems committee: Ben Burton, Bernard Blackham, David Clark, David Kennedy, Andrew Gray, Katherine Kavanagh, David Ananian-Cooper, Dmitry Kamenetsky, Robert Newey, Christopher Chen, Judith Heglers and Mike Clapper. Their ability to construct new and interesting problems continues to astonish me. Thanks are also due to Margot Phillipps, Terry McClelland, Jan Honnens and Katherine Kavanagh: teachers who have provided second round moderation of the competition. Their comments and suggestions have helped to ensure that the problems are understandable by the students and accessible to them.

Special thanks are due to Bernadette Webster for careful proofreading of this book, and of all of the CAT problems before they go to print, and to Heather Sommariva for the cover design.

This book was typeset in \textsc{\LaTeX}, the mathematical typesetting system designed by Donald Knuth. Many contributors have given freely of their time and expertise to extend the capabilities of \textsc{\LaTeX}. The TikZ/PGF package written by Till Tantau was used in all of the diagrams in this book. Finally, special thanks are due to the late Ian Lisle for expert advice on all things \textsc{\LaTeX}.

Mike Clapper
Executive Director
Australian Mathematics Trust
September 2016
Questions

Your time machine is playing up. How do you respond?
Applying Rules

1. **Zabs**

On a distant planet, the dominant carnivore, the zab, is nearing extinction. The number of zabs born in any year is one more than the (positive) difference between the number born in the previous year and the number born in the year before that.

Examples: If 7 zabs were born last year and 5 the year before, 3 would be born this year. If 7 zabs were born last year and 10 the year before, 4 would be born this year.

2 zabs were born in the year 2000 and 9 zabs were born in 2001. What is the first year after 2000 when just 1 zab will be born?

(A) 2009  
(B) 2011  
(C) 2013  
(D) 2015  
(E) 2017

2. **Word Search**

A word search uses the following special symbols:

- ? represents a single letter
- * represents any number of letters, including no letters.

In order for a search term to match a word, it must represent the entire word from start to finish. For example, b*t matches bat but not bath.

How many of the following words does b??st*ing match?

- blasting
- blustering
- boasting
- boosting
- bootstrapping
- bowstrings
- bristling
- busting

(A) 2  
(B) 4  
(C) 5  
(D) 6  
(E) 7