BAPC 2008

Program Booklet

TU Delft

W.I.S.V.
'Christiaan Huygens'

acm International Collegiate Programming Contest

IBM event sponsor

Colophon

This program is a single production by the CHipCie, a committee of W.I.S.V. 'Christiaan Huygens' in Delft, for the purpose of the Benelux Algorithm Programming Contest 2008, held on the 25th of October 2008 at the Delft University of Technology.

Texts	: Thomas Verwoerd
	Boaz Pat-El
	Robin van den Berg
	Raul Kooter
	Thomas Schaap
Logo	: Joey van den Heuvel
Layout	: Robin van den Berg

This booklet was printed by Drukwerkdeal.nl.
All information in this booklet is subject to change. Check the website for the most recent information.

If you have any questions, please contact the CHipCie:
W.I.S.V. 'Christiaan Huygens'
Mekelweg 4
2628 CD Delft

Contact person	: Thomas Verwoerd
Telephone	: +31 (0)15-278 2532
Fax	: +31 (0)15-278 2690
E-mail	: chipcie@ch.tudelft.nl
Website	: http://www.bapc.eu/

Preface

“You can learn how to program in Java or Pascal” is a song I learned in my first year. I was young and naïve and convinced that every problem was easily solvable in any programming language. But after a few months of learning to program, I found out there is more to it than just bluntly writing code in a Notepad window and compiling it. I came to see a good comprehension of the problem was required to be able to write an efficient algorithm.

Six years wiser, I have the honor of leading the contest that challenges the skills of writing efficient algorithms and solving problems of students from the Benelux. You are reading the program booklet of the Benelux Algorithm Programming Contest 2008 organised at the Delft University of Technology.

For some people the Benelux Algorithm Programming Contest is the first step towards the ICPC World Finals. Others might already have proven themselves as the best of their universities at their local qualifying rounds. For the best contestants the next step will be the North Western European Regional Contest (NWERC) in Utrecht. Although some universities use this contest as a selection for the NWERC, it will be a good practice and a chance to win some nice prices for everyone.

This booklet provides information about the Benelux Algorithm Programming Contest. Additional information and updates can be found on the website, http://www.bapc.eu, and the organisation can be contacted any time at chipcie@ch.tudelft.nl.

I wish everybody the best of luck with the preparations and may the best team win.

Thomas Verwoerd
Contest Director BAPC 2008

Contents

TU Delft
TU Delft
Mekelweg 4
Faculteit
Elektrotechniek
Wiskunde en
Informatica

Route Description

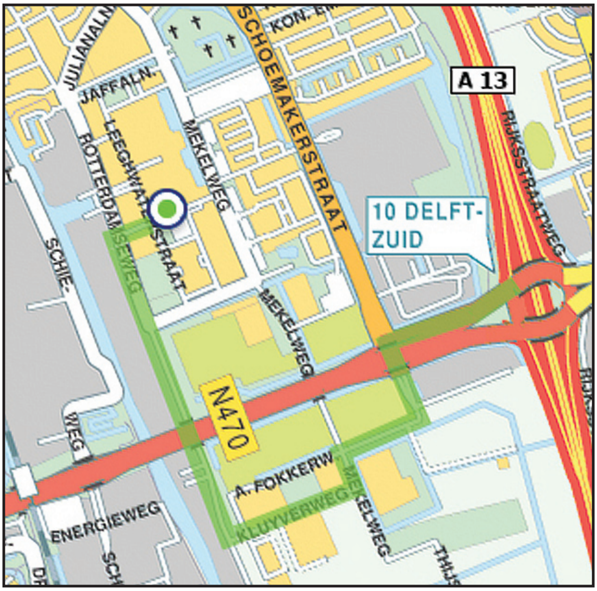
Location

Faculty of EEMCS
Cornelis Drebbelweg 5
2628 CM Delft

By public transport

By train, travel to Delft Central Station. From there take bus 129 to Rotterdam. Get off at bus stop "Drebbelweg". For a detailed train and bus schedule, visit http://www.9292ov.nl or http://www.connexxion.nl.

By car



Because of the reconstruction of the Mekelpark you will probably have to take a detour: follow the yellow sign to "Elektotechniek". The faculty of EEMCS is best accessible by car from highway A13. From both directions take exit 10 Delft Zuid/TU Delft. Immediately take the exit toward Schoemakerstraat and turn left at the end. Follow the road turning right to Watermanweg. At the end of this road, turn left and take the first right (Kluyverweg). At the end turn right to the Rotterdamseweg. After 1.1 kilometers turn right to the Cornelis Drebbelweg. Continue until Building 35 appears to your left.

Day Program

Time	Activity
10:00	Team registration; coffee and tea
10:30	Welcoming speech and instructions
11:00	Start of test session
11:45	End of test session; lunch
12:30	Last minute remarks
13:00	Start of contest
18:00	End of contest; drinks
18:30	Award ceremony
20:00	Dinner

Example Problem

Euro Efficiency

From: NWERC 2002

On January the 1st 2002, The Netherlands, and several other European countries abandoned their national currency in favour of the Euro. This changed the ease of paying, and not just internationally.

A student buying a 68 guilder book before January the 1st could pay for the book with one 50 guilder banknote and two 10 guilder banknotes, receiving two guilders in change. In short: 50 + 10 + 10 - 1 – 1 = 68. Other ways of paying were: 50 + 25 - 5 - 1 - 1, or 100 - 25 - 5 - 1 – 1. Either way, there are always 5 units (banknotes or coins) involved in the payment process, and it could not be done with less than 5 units.

Buying a 68 Euro book is easier these days: 50 + 20 – 2 = 68, so only 3 units are involved. This is no coincidence; in many other cases paying with euros is more efficient than paying with guilders. On average the Euro is more efficient. This has nothing to do, of course, with the value of the Euro, but with the units chosen. The units for guilders used to be: 1, 2.5, 5, 10, 25, 50, whereas the units for the Euro are: 1, 2, 5, 10, 20, 50.

For this problem we restrict ourselves to amounts up to 100 cents. The Euro has coins with values 1, 2, 5, 10, 20, 50 eurocents. In paying an arbitrary amount in the range [1, 100] eurocents, on average 2.96 coins are involved, either as payment or as change. The Euro series is not optimal in this sense. With coins 1, 24, 34, 39, 46, 50 an amount of 68 cents can be paid using two coins. The average number of coins involved in paying an amount in the range [1, 100] is 2.52.

Calculations with the latter series are more complex, however. That is, mental calculations. These calculations could easily be programmed in any mobile phone, which nearly everybody carries around nowadays. Preparing for the future, a committee of the European Central Bank is studying the efficiency of series of coins, to find the most efficient series for amounts up to 100 eurocents. They need your help.

Problem

Write a program that, given a series of coins, calculates the average and maximum number of coins needed to pay any amount up to and including 100 cents. You may assume that both parties involved have sufficient numbers of any coin at their disposal.

Input

The first line of the input contains the number of test cases. Each test case is described by 6 different positive integers on a single line: the values of the coins, in ascending order. The first number is always 1. The last number is less than 100.

Output

For each test case the output is a single line containing first the average and then the maximum number of coins involved in paying an amount in the range [1, 100]. These values are separated by a space. As in the example, the average should always contain two digits behind the decimal point. The maximum is always an integer.

Example

Input

```
3
1 2 5 10 20 50
1 24 34 39 46 50
1 2 3 7 19 72
```

Output

```
2.96 5
2.52 3
2.80 4
```

Systems & Tools

By Jeroen Dekkers

During the contest the computer is your most valuable resource (except maybe for coffee). Some explanation about what you can expect from our systems is therefore in order.

Hardware

During the contest each team will have exactly one computer (with TFT screen, keyboard and mouse). Because of possible revisions by the faculty, the specification of the computers are unknown at the time of the publication of this booklet, but they will vary between two years old and brand new. In any case, the jury will use similar machines to judge your submissions.

Software

The only operating system available to you will be *Debian GNU/Linux*. The jury will run your submissions on the same system with the same compilers. During the contest you will be able to choose between the languages C, C++ or Java. Using different languages for different solutions is possible and allowed.

Versions

The machines will be based on the *Lenny* release of Debian GNU/Linux.

Window Managers

Available window managers are:

- GNOME 2.22 (default)
- KDE 3.5
- Xfce 4.4
- Ion3 20080707

Compilers

Available compilers are:

- GCC 4.3.1 (C and C++)
- SUN Java 6 update 7

Editors

Available editors are (among others):

- Emacs 22.2
- Vim 7.1.314
- Eclipse 3.2.2
- Netbeans 6.0.1
- Nano 2.0.7
- Kate 3.5

Disclaimer

Given that this booklet is printed a long time before the BAPC takes places, the provided list is only an indication and is likely to change.

An an up-to-date list is provided on our website. If your favourite window manager or editor is not on the list or if you have other requirements, you can send us a request and we will look whether it is possible to install it.

```
>[illegible]
I DON'T KNOW THAT WORD.
>MY NAMEIS COLPANTS
I DON'T KNOW THAT WORD.
>FGDHUDAURULHWFA
I DON'T KNOW THAT WORD.
>CHRISQ
I DON'T KNOW THAT WORD.
```

Hunting-guild 'The Judges'

Months before you received this booklet a few people were already exploring the deepest parts of the jungles that are their imagination, hunting for interesting problems from all walks of life. Many a prey was shot during the hunt and the finest specimens have been selected and stuffed for you to see.

Although the hunt for problems is an exciting one, especially when meeting a cunning problem, the hunt for a good team on a contest can't beat it. The hunt for problems is just preparation to find the best bait with which to lure the best teams into our carefully crafted trap.

But even during the contest we won't put our sharp senses to rest. While you are trying hard to avoid our smaller traps in order to reach the big one, we are still hunting. Hunting for errors in the automated judging of your problems. Even though we are infallible, these computers that automatically judge your solutions may not be. Thus it might even occur that an erroneous judgment comes within our sight and needs correction. Even though this might not seem very nice of us, we are strict and fair in our judgments and will correct errors if needed.

And we'll be hunting for your questions. With every question we receive we will hunt the issue down and then shoot the question, replying you should "read the problem specification more carefully." For we are infallible and your question thus can't possibly stem from any ambiguities, errors or other unclarities left in the exercises. In the remote case that, despite our infallibility, an imperfection might have slipped past our nets, we will make sure all get to know about it, not just you.

Every time we go hunting we are thrilled by the experience. Will we catch something today? During the contest I hope we won't find much prey: the less we need to do, the smoother things go. I hope you will also experience the thrill as you hunt for the solutions within the limited time of your hunt.

Good hunting!

On behalf of the judges,

Thomas Schaap, Head of Judges

Rules & Regulations

1. Definitions

TU Delft	Delft University of Technology
BAPC	The 2008 Benelux Algorithm Programming Contest, which takes place on 25 October 2008 at the faculty of Electrical Engineering, Mathematics and Computer Science of TU Delft.
CH	W.I.S.V. `Christiaan Huygens', Study Association for the study programs Mathematics and Computer Science of the TU Delft.
Organisation	The members of the organizing committee of CH.
Website	The website, maintained by the organisation and available at http://www.bapc.eu
Jury	The group of people responsible for making the problems and checking the solutions submitted by the participants.
Runners	By the organisation appointed persons, responsible for delivering print-outs, answering questions and various other tasks.
Crew	Organisation, members of the jury or runners.
Participant	Member of a participating team that competes in the BAPC.
Submission	A submission of a solution by a team.

2. Organisation

2.1 The organisation exists of members of CH.

2.2 The organisation has formed a jury which exists of people of the organisation, students and staff of the TU Delft and other universities.

2.3 The organisation has appointed some runners who will watch over the competition areas during the contest, hand out the print-outs and balloons and will be available for practical questions during the day.

2.4 All staff will be recognizable by their shirt and/or badge.

2.5 The organisation will form a board of contest leaders.

3. Participation

3.1 Introduction

3.1.1 Participation is only possible in teams consisting of up to 3 persons.

3.1.2 There are two pools: One for student teams and one for business teams.

3.1.3 Changing the composition of a team is only possible when the organisation has agreed upon this.

3.1.4 The organisation shall decide how many teams from each institution shall be allowed to compete. The organisation will consider the number of interested contestants from each institution.

3.1.5 The organisation has the right to deny teams of participation before the start of the contest.

3.2 Student teams

A student team:

3.2.1 may participate for free.

3.2.2 exists of students from the same institution and who are not participating in another team.

3.2.3 has a coach, which is the contact person of a team. This can be a team member or a student or staff member of the institution.

3.2.4 participates in the student teams pool for the title "Benelux Champion Algorithm Programming 2008" with the cup and the prize money of 1024,- 512,- and 256,- euro for first, second, and third places respectively.

3.2.5 consists of students from the same institution.

3.2.6 consists of students who are eligible for the North Western European Programming Contest 2008.

3.3 Business teams

A business team:

3.3.1 pays the registration fee, before the start of the contest.

3.3.2 consists out of persons who are employed by the same company or institution.

3.3.3 Compete in the pool business teams for the title "Benelux Champion Algorithm Programming 2008" and the prize money of 512 euro.

4. BAPC

4.1 Introduction

4.1.1 The language used on BAPC is English.

4.1.2 BAPC lasts for 5 hours.

4.1.3 From the beginning until one hour before the end of the BAPC, the scores are displayed.

4.2 Problems

4.2.1 The jury will provide at least 6 and at most 10 problems.

4.2.2 When a problem is unclear a "clarification request" can be sent to the jury. The jury will respond to this request. If the response is relevant to all teams, the jury will send the response to all teams.

4.2.3 The jury has the right to change or withdraw problems during the contest. When this happens the jury will inform all teams.

4.3 System

4.3.1 Each team has the same workplace available.

4.3.2 A solution has to be written in C/C++ or Java.

4.3.3 The jury decides per programming language which libraries and function calls are allowed to be used in the solutions.

4.3.4 All prints made by the teams are brought by a runner. Participants are not allowed to be near the printers.

4.3.5 A team is allowed to bring up to 30 A4-sized pages of documentation; no other documentation (including books and manuals) is allowed.

4.3.6 A team is not allowed to bring software or hardware on the contest floor.

4.4 Department rules

4.4.1 Inside the building where the BAPC takes place, the house rules also apply.

4.4.2 Inside computer rooms eating, drinking, and smoking is not allowed.

4.4.3 The use of hardware, including all calculators, which is not approved by the organisation is forbidden, with exceptions of simple watches and medical equipment.

4.4.4 Changing of hardware or Operating software is strictly forbidden.

4.4.5 During the contesg, communication within the team and crew is allowed. Communication with everyone else is forbidden during the contest.

4.4.6 Participants will follow orders given by the crew.

4.4.7 Participants will wear the shirt and badge provided by the organisation.

4.5 Judgment

4.5.1 A submission is handled by an automated jury system. The organisation is responsible for behavior of the system. The jury will check the behavior of the system.

4.5.2 Each submission is acknowledged.

4.5.3 For each problem, the jury has a correct solution and test data.

4.5.4 A submission is correct when it has a solution to the input in a time limit decided by the jury and the output is the same as the output of the jury. This time limit is not announced to the teams.

4.5.5 The winner of a pool is decided by (in order):

1. The team with the most correctly solved problems.
2. The team with the least solving time. This is the sum of the time needed for every solved problem, plus a 20-minute penalty for each wrong submission until the first correct submission. (Incorrect solutions for which a team has not submitted a correct solution or incorrect solutions submitted after a correct solution was accepted do not add to the solving time.)

4.5.6 The jury is responsible for everything that has to do with the problem set and can be contacted for this through the “clarification requests.”

5. Special rules

5.1 The organisation has the right to disqualify teams for misbehavior or breaking the rules, including dislodging extension cords, unauthorized modification of contest materials, or distracting behavior.

5.2 The contest leaders have the right to stop the contest, extend the contest time, temporarily block submissions for all teams or change the scores in exceptional conditions.

5.3 In situations to which no rule applies, the organisation decides.

Tips & Tricks

By Boaz Pat-El

How well a team performs during a contest is a function of each member's math, programming and co-operative skills. How you should tackle the problems is something that differs per team and therefore is something that you should work out for yourselves.

However, as jury we noticed that many teams make some common mistakes. Here we illustrate some of these mistakes and expect that, if you took the effort to read this page, you will avoid making them during the contest.

Check the output specification

In a programming contest, every second counts. Because of this, some teams are prone to submit their solutions in such a hurry that they forget to check whether their output corresponds to the specification of the problem. Results include, but are not limited to, redundant (or lack of) commas, spaces or newlines, as well as debug-information being present in the output.

$O(n^5)$ solutions usually don't work

It's nice that you found a solution to a problem. Still, check the maximum input size of the problem before submitting. If sufficiently large, your approach of going through the whole problem-space will probably take up too much time and will be rejected.

Take edge cases into account

This is a no-brainer right? Even so, you would be surprised at how many submissions solve only half of the input-cases, or give the correct output for all but one case.

Organisation

Thomas Verwoerd
Chairman

Boaz Pat-El
Secretary, Judge

Robin van den Berg
Treasurer

Thomas Schaap
Head of Judges

Raul Kooter
Public Relations

Jeroen Dekkers
Systems

Martijn van Oosterhout
Systems

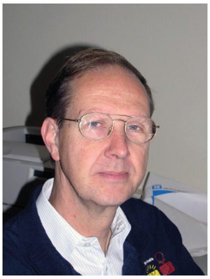
Martin van Buuren
Organisation

Mark Janssen
Organisation

Arnout Boks
Judge

Cynthia Liem
Judge

Michiel de Reus
Qualitate Qua


Dr. P.G. Kluit
Judge, Legend

External Judges:

Eljakim Schrijvers

Boris de Wilde

David Koh

Special Thanks

The CHipCie would like to thank everybody who made the Benelux Algorithm Programming Contest 2008 possible.

W.I.S.V.
'Christiaan Huygens'

TU Delft

acm International Collegiate Programming Contest
IBM event sponsor

Prof.dr. Daan Lenstra
Wim Haan
John Veltema
René Pingen
Joey van den Heuvel
Maarten van der Beek

The Runners

The Jury, with special regard to:
Peter Kluit
David Koh
Kim Schrijvers
Boris de Wilde

Our Sponsors:
Technolution,
ASML, Atos Origin, Logica, NCIM, Ortec, Saen Options

Benelux
Algorithm
Programming
Contest
2008

Brought to you by:

Premium sponsor:

Technolution
AUTOMATION TECHNOLOGY

Sponsors:

NCIM GROEP

Atos Origin™

Releasing your potential
logica

ASML

ORTEC
PROFESSIONALS IN PLANNING

Program Booklet

TU Delft

W.I.S.V.
'Christiaan Huygens'

acm International Collegiate Programming Contest

IBM event sponsor