

DAPC 2023 Training Sessions

Session 1

Verwoerd

September 9, 2023

Introduction

Welcome

- Welcome to the DAPC 2023 Training Sessions
- 4 sessions
- We will discuss all last years problems of the DAPC and BAPC
- Every session some problems we solve together
- Other problems you can solve in between sessions and only the solutions will be presented
- Every session starts with some practical information
- Maybe guest speakers?

Who am I

- Alumnus, working in the Software Industry
- Involved in organizing programming contests since 2003 as volunteer
- “Coach” for TU Delft teams since NWERC 2003
- Twice coach on the World Finals

This work is licensed under a Creative Commons
“Attribution-ShareAlike 4.0 International” license.



Session 1 (Today)

- Introduction to Programming Contests
- Reading a problem
- Introduction to DOMJudge
- Some tips on estimate the problem complexity
- Solving an ad-hoc Math problem
- Meet and Greet to look for team or team-members

- Team Tactics
- Utilizing the Test Session
- How to select problems
- Dealing with wrong submissions
- Solutions to the Ad-hoc and Math Problems
- Solving Sorting and Search Problems

- Creating a Team Reference Document
- Solutions to Sorting and Search Problems
- Solving Interactive Problems and Randomized Input Problems

- Role of the coach on big contests
- Tips, tricks and common mistakes
- Solutions to the Interactive Problems and Randomized Input Problems
- Solving the Hardest Problems

Introduction to Programming Contests

What is a programming contest?

- Team of 3 people
- Single computer
- Solve as many problems from the problem set (8 to 15 problems)
- In 5 hours
- In any order

What is a programming contest?

- Team of 3 people
- Single computer
- Solve as many problems from the problem set (8 to 15 problems)
- In 5 hours
- In any order
 - Solve it efficiently
 - do it as quickly as possible (under pressure)
 - and do it correctly (without bugs)

What is a programming contest?

- Team of 3 people
- Single computer
- Solve as many problems from the problem set (8 to 15 problems)
- In 5 hours
- In any order
 - Solve it efficiently
 - do it as quickly as possible (under pressure)
 - and do it correctly (without bugs)
- With limited documentation and no internet

How is score calculated?

- Sorted by number of problems solved

How is score calculated?

- Sorted by number of problems solved
- Sorted by the total time for solved problems

How is score calculated?

- Sorted by number of problems solved
- Sorted by the total time for solved problems
 - Time in minutes since the start of the contest
 - Penalty for each wrong attempt on a solved solution of 20 minutes












How is score calculated?

- Sorted by number of problems solved
- Sorted by the total time for solved problems
 - Time in minutes since the start of the contest
 - Penalty for each wrong attempt on a solved solution of 20 minutes
 - Penalty time is counts only if the problem is solved afterward.
 - Penalty time does not reduce your contest time.
 - Penalty time is not added after wrong attempts after the problem is solved.
 - No penalty for compiler errors.

Example Scoreboard

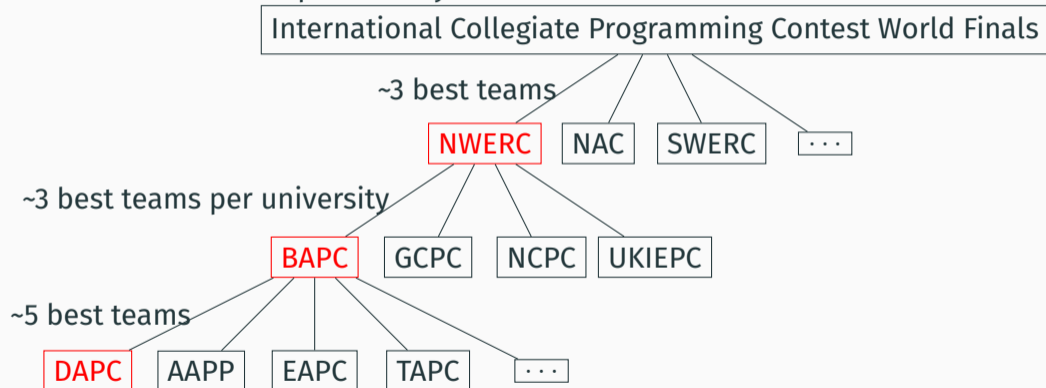
DAPC 2022

final standings

RANK	TEAM	SCORE	A	B	C	D	E	F	G	H	I	J	K	L
1	 Segfault go BRRRR Delft University of Technology	12 1090	22 1 try	48 2 tries	41 1 try	94 1 try	66 1 try	7 1 try	190 3 tries	223 1 try	118 1 try	106 2 tries	85 1 try	10 1 try
2	 ADA Refactor Delft University of Technology	11 1087	21 2 tries	44 2 tries	98 1 try	108 1 try	77 1 try	25 1 try	251 1 try		66 1 try	162 4 tries	129 1 try	6 1 try
3	 Chindia Targoviste Delft University of Technology	11 1273	74 3 tries	31 1 try	188 1 try	112 1 try	61 1 try	14 1 try	226 1 try		142 2 tries	149 2 tries	175 1 try	21 1 try
4	 WA & Chill Delft University of Technology	11 1424	9 1 try	67 1 try	111 1 try	181 1 try	131 4 tries	35 1 try	298 3 tries		154 1 try	102 3 tries	143 1 try	53 1 try
5	 Placeholder Delft University of Technology	11 1589	50 3 tries	110 2 tries	76 1 try	181 1 try	224 1 try	19 1 try	289 4 tries		144 1 try	99 2 tries	160 1 try	17 5 tries
6	 Exponential Fenwick Delft University of Technology	10 749	14 1 try	62 2 tries	139 4 tries	94 1 try	61 1 try	18 1 try			42 2 tries	99 1 try	116 1 try	4 1 try
7	 Dirty Bits Done Dirt Cheap Delft University of Technology	10 1198	11 1 try	35 2 tries	64 1 try	129 1 try	191 1 try	22 1 try			147 1 try	264 10 tries	105 1 try	30 1 try
8	 Sleetje3 Delft University of Technology	10 1311	30 1 try	94 2 tries	49 1 try	222 1 try	209 2 tries	19 1 try			254 1 try	169 5 tries	79 1 try	66 1 try
9	 SMG Delft University of Technology	10 1534	48 1 try	10 1 try	201 3 tries	169 1 try	219 3 tries	21 1 try			267 8 tries	93 2 tries	192 4 tries	14 1 try
10	 Poland Mountain Delft University of Technology	10 1626	65 3 tries	55 1 try	287 4 tries	217 1 try	119 1 try	20 1 try			253 1 try	172 2 tries	218 1 try	100 1 try
11	 NoDucksGiven Delft University of Technology	10 1826	128 3 tries	147 1 try	73 1 try	257 1 try	234 6 tries	36 1 try	290 4 tries			175 2 tries	160 1 try	26 5 tries
12	 bits by dre Delft University of Technology	9 1396	24 1 try	55 1 try	132 1 try	246 1 try	262 1 try	52 1 try				221 3 tries	234 3 tries	90 1 try

Road to the world finals

The DAPC is an official preliminary of the ICPC.



Reading a problem

Problem structure

A typical problem has the following structure

- Problem description
- Input description
- Output description
- Example input/output
- A time limit in seconds

You are asked to write a program that solves the problem for all valid inputs within the time limit.

Example problem

Problem description

Write a program that multiplies pairs of integers.

Input description

The input consists of:

- One line with an integer t ($1 \leq t \leq 100$), the number of test cases.
- t lines, each with two integers a and b ($|a|, |b| \leq 10^6$), the numbers to multiply.

Output description

For each test case, output the value of $a \times b$.

Example problem

Sample input	Sample output
4	12
3 4	0
13 0	8
1 8	10000
100 100	

Solution in C++

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int t;
6      cin >> t;
7      for (int i = 0; i < t; i++) {
8          int a, b;
9          cin >> a >> b;
10         cout << a * b << endl;
11     }
12     return 0;
13 }
```

Solution in Java

```
1 import java.io.*;
2
3 class Problem {
4     public static void main(String[] args) throws IOException {
5         var input = new BufferedReader(new InputStreamReader(System.in));
6         var cases = Integer.parseInt(input.readLine());
7         for (int i = 0; i < cases; i++) {
8             var line = input.readLine().split(" ");
9             System.out.println(
10                 Integer.parseInt(line[0]) * Integer.parseInt(line[1])
11             );
12         }
13     }
14 }
```

Solution in Kotlin and Python

```
1 fun main() {
2     val t = readln().toInt();
3     System.`in`.bufferedReader().lineSequence().take(t).forEach { line ->
4         println(line.split(" ").map { it.toInt() }.let { (a, b) -> a * b })
5     }
6 }
```

```
1 t = int(input())
2 for t in range(t):
3     numbers = list(map(int, input().split()))
4     print(numbers[0] * numbers[1])
```

Introduction to DOMJudge

Submitting the Solution

- During the contest you submit to a contest control system
 - Usually DOMJudge, but sometimes Kattis or PC²
- Submit solutions
- Ask questions about the problems or programming environment
- Read clarifications from the jury

Domjudge Interface - home

RANK	TEAM	SCORE	TEST
1	Coach	0	0

Submissions

No submissions

Clarifications

No clarifications.

Clarification Requests

No clarification request.

[request clarification](#)

Domjudge Interface - problems

The screenshot displays the Domjudge web interface. At the top, a dark navigation bar contains the site name 'DOMjudge' and links for 'Home', 'Problemset', 'Print', and 'Scoreboard'. On the right side of this bar are buttons for 'Submit', 'Logout', and a 'Training' dropdown menu, along with a clock showing '123d 37:01'. The main content area features the heading 'Trainging problems' (note the typo). Below this, a problem card is shown with a 'test' button, the title 'Number List', and the text 'Limits: 1 second / 2 GB'. Two rows of small square icons represent the problem's progress. At the bottom of the card are 'samples' and 'Submit' buttons.

Domjudge Interface - submit

The screenshot displays the Domjudge web interface. At the top, there is a navigation bar with links for Home, Problemset, Print, and Scoreboard. On the right side of the navigation bar, there are buttons for Submit, Logout, and Training, along with a clock showing 12:3d 32:33. The main content area is divided into several sections: Submissions (with a 'No submissions' message), Clarifications, and Clarification Requests. A 'Submit' modal form is open in the center, containing the following fields and options:

- Source files:** A text input field containing 'example.kt' and a 'Browse' button.
- Problem:** A dropdown menu with 'test - Number List' selected.
- Language:** A dropdown menu with 'Kotlin' selected.
- Main class:** A text input field containing 'ExampleKt'.
- Footer:** A note stating 'The entry point for your code.' and two buttons: 'Cancel' and 'Submit'.

Are the solutions correct?

RANK	TEAM	SCORE	TEST
1	Coach	0 0	2 = 4 test

Submission done! Watch for the verdict in the list below.

x

Submissions

time	problem	lang	result
15:43	TEST	PY3	PENDING
15:42	TEST	JAVA	PENDING
15:42	TEST	CPP	PENDING
15:42	TEST	KT	PENDING

Clarifications

No clarifications.

Clarification Requests

No clarification request.

[request clarification](#)

We made a whoopsy?

RANK	TEAM	SCORE	TEST
1	Coach	0 0	3/1 tries

Submission done! Watch for the verdict in the list below.

x

Submissions

time	problem	lang	result
15:43	TEST	PY3	PENDING
15:42	TEST	JAVA	WRONG-ANSWER
15:42	TEST	CPP	WRONG-ANSWER
15:42	TEST	KT	WRONG-ANSWER

Clarifications

No clarifications.

Clarification Requests

No clarification request.

request clarification

RANK	TEAM	SCORE	TEST
1	Coach	1 / 89	29 4 files

Submission done! Watch for the verdict in the list below.

X

Submissions

time	problem	lang	result
15:43	TEST	PY3	CORRECT
15:42	TEST	JAVA	WRONG-ANSWER
15:42	TEST	CPP	WRONG-ANSWER
15:42	TEST	KT	WRONG-ANSWER

Clarifications

No clarifications.

Clarification Requests

No clarification request.

request clarification

Lets ask the jury

The screenshot shows the DOMjudge web interface. At the top, there is a navigation bar with 'DOMjudge', 'Home', 'Problems', 'Print', and 'Scoreboard'. On the right, there are buttons for 'Submit', 'Logout', and 'Training', along with a clock showing '12:3d 23:18'. The main content area is divided into three sections: 'Submissions', 'Clarifications', and 'Clarification Requests'. The 'Submissions' section contains a table with the following data:

time	problem	lang
15:43	TEST	PV3
15:42	TEST	JAVA
15:42	TEST	CPP
15:42	TEST	KT

A modal dialog box titled 'Send clarification request' is open in the center. It has a close button (X) in the top right corner. The dialog contains the following fields:

- Recipient:** A dropdown menu with 'Jury' selected.
- Subject:** A dropdown menu with 'test: Number List' selected.
- Message:** A text area containing the text: 'Why did the first 3 submissions fail? They do the same as the accepted one.'

At the bottom of the dialog, there are two buttons: 'Cancel' and 'Send'.

Lets hope they respond fast

RANK	TEAM	SCORE	TEST
1	Coach	1 89	29 4 times

Clarification sent to the jury



Submissions

time	problem	lang	result
15:43	TEST	PY3	CORRECT
15:42	TEST	JAVA	WRONG-ANSWER
15:42	TEST	CPP	WRONG-ANSWER
15:42	TEST	KT	WRONG-ANSWER

Clarifications

No clarifications.

Clarification Requests

time	from	to	subject	text
15:52	Coach	Jury	problem test	Why did the first 3 submissions fail? They do the same as the accepted one.

[request clarification](#)

We have a response

RANK	TEAM	SCORE	TEST
1	Coach	1 / 89	29 4 tests

Submissions			
time	problem	lang	result
15:43	TEST	PY3	CORRECT
15:42	TEST	JAVA	WRONG-ANSWER
15:42	TEST	CPP	WRONG-ANSWER
15:42	TEST	KT	WRONG-ANSWER

Clarifications					
time	from	to	subject	text	
15:53	Jury	Coach	problem test	No comment.	

Clarification Requests					
time	from	to	subject	text	
15:52	Coach	Jury	problem test	Why did the first 3 submissions fail? They do the same as the accepted one.	

[request clarification](#)

The jury is not helping us

DOMjudge Home Problemset Print Scoreboard

Submit Logout Training 123d 19:03

Submissions

time	problem	lang
15:43	TEST	PY3
15:42	TEST	JAVA
15:42	TEST	CPP
15:42	TEST	KT

Clarification Request

Subject: Problem test: Number List 15:52

From: Coach (t3) To: Jury

Why did the first 3 submissions fail? They do the same as the accepted one.

Subject: Problem test: Number List 15:53

From: Jury To: Coach (t3)

> Why did the first 3 submissions fail? They do the same as the accepted one.
No comment.

reply to this clarification Close

Clarifications

subject	text
problem test	No comment.

Clarification Requests

Why did the first 3 submissions fail? They do the same as the accepted one.

Why did the 3 solutions fail?

- Lets check the input again: $|a|, |b| \leq 10^6$

Why did the 3 solutions fail?

- Lets check the input again: $|a|, |b| \leq 10^6$
- Worst case scenario: $a = 10^6$ and $b = 10^6$ giving $a \times b = 10^{12}$

Why did the 3 solutions fail?

- Lets check the input again: $|a|, |b| \leq 10^6$
- Worst case scenario: $a = 10^6$ and $b = 10^6$ giving $a \times b = 10^{12}$
- Does 10^{12} fit in a 32-bit int?

Why did the 3 solutions fail?

- Lets check the input again: $|a|, |b| \leq 10^6$
- Worst case scenario: $a = 10^6$ and $b = 10^6$ giving $a \times b = 10^{12}$
- Does 10^{12} fit in a 32-bit int?
- $\log_2 10^{12} \approx 40$, so **NO**, 40 bits don't fit in an int

Why did the 3 solutions fail?

- Lets check the input again: $|a|, |b| \leq 10^6$
- Worst case scenario: $a = 10^6$ and $b = 10^6$ giving $a \times b = 10^{12}$
- Does 10^{12} fit in a 32-bit int?
- $\log_2 10^{12} \approx 40$, so **NO**, 40 bits don't fit in an int
- Use `long` (`long`) when possible, except in Python

Solution in C++

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int t;
6      cin >> t;
7      for (int i = 0; i < t; i++) {
8          long long a, b;
9          cin >> a >> b;
10         cout << a * b << endl;
11     }
12     return 0;
13 }
```

Solution in Java

```
1 import java.io.*;
2
3 class ProblemCorrect {
4     public static void main(String[] args) throws IOException {
5         var input = new BufferedReader(new InputStreamReader(System.in));
6         var cases = Integer.parseInt(input.readLine());
7         for (int i = 0; i < cases; i++) {
8             var line = input.readLine().split(" ");
9             System.out.println(
10                 Long.parseLong(line[0]) * Long.parseLong(line[1])
11             );
12         }
13     }
14 }
```

Solution in Kotlin

```
1 fun main() {
2     val t = readln().toInt();
3     System.`in`.bufferedReader().lineSequence().take(t).forEach { line ->
4         println(line.split(" ").map { it.toLong() }.let { (a, b) -> a * b })
5     }
6 }
```

All solutions correct

RANK	TEAM	SCORE	TEST
1	Coach	1 / 89	29 4 tests

Submissions			
time	problem	lang	result
16:15	TEST	KT	CORRECT
16:15	TEST	JAVA	CORRECT
16:14	TEST	CPP	CORRECT
15:43	TEST	PY3	CORRECT
15:42	TEST	JAVA	WRONG-ANSWER
15:42	TEST	CPP	WRONG-ANSWER
15:42	TEST	KT	WRONG-ANSWER

Clarifications				
time	from	to	subject	text
15:53	Jury	Coach	problem test	No comment.

Clarification Requests				
time	from	to	subject	text
15:52	Coach	Jury	problem test	Why did the first 3 submissions fail? They do the same as the accepted one.

[request clarification](#)

Estimating problem complexity

About time limit

- The time limit specifies the time your program may run
- This includes JVM-startup and I/O
- High time limit signifies
 - lots of I/O
 - Slower algorithms can be accepted
- Low limit signifies fast algorithms, usually the use of formulas
- You can use the time limit to check your code on your local machine

```
$ time myjava ProblemA < worst-case.in
```


About input size¹

Based on the input size you can get an idea of the time complexity.

$\mathcal{O}(n!)$	$n \leq 10$	$\mathcal{O}(n \log^2 n)$	$n \leq 10^5$
$\mathcal{O}(2^n)$	$n \leq 20$	$\mathcal{O}(n \log n)$	$n \leq 10^6$
$\mathcal{O}(n^3)$	$n \leq 500$	$\mathcal{O}(n)$	$n \leq 10^8$
$\mathcal{O}(n^2 \log n)$	$n \leq 1000$	$\mathcal{O}(\sqrt{n})$	$n \leq 10^{15}$
$\mathcal{O}(n^2)$	$n \leq 5000$	$\mathcal{O}(\log n)$	$n \leq 10^{18}$
$\mathcal{O}(n\sqrt{n})$	$n \leq 10^5$		

Warning: This is not guaranteed to be always the case!

¹<https://gcpc.nwerc.eu/primer.pdf>

Solving an ad-hoc math problem

An other problem

- Source BAPC Preliminaries 2022
- Problem name: Fastestest Function
- Time limit: 1s

Original problem written by the BAPC 2022 jury and licensed under Creative Commons Attribution-ShareAlike 4.0 International.



Problem: Fastestest Function

You are working as a software developer for the Bug Acquisition Programming Company. They developed a specific piece of software called Program C that they sell to their clients. For the past weeks, you have been working on optimising a specific function `foo` in the main code path in Program C. You have made it a lot faster and would like to show off to your boss about it.

Your IDE has a nice tool that allows you to profile your code and tell you what percentage of the total running time `foo` takes. You can run this on the version before your change and after your change. However, you think it looks a lot cooler if you can just tell your boss how much faster you have made `foo` itself.

Problem: Fastestest Function: Input and Output

Input

The input consists of:

- One line with two integers x and y ($0 < x, y < 100$), where x is the percentage of the total running time that `foo` took before optimising and y the percentage of the total running time it took after optimising.

Output

Output the factor of how much faster `foo` got after your optimization.

Your answer should have an absolute or relative error of at most 10^{-6} .

Problem: Fastestest Function: Samples

Sample Input 1	Sample Output 1
75 50	3.0

So foo first took 75% of the total running time, after optimization only 50% of the running time. foo is now 3× faster than before.

Sample Input 2	Sample Output 2
50 75	0.3333333333333333

Sample Input 3	Sample Output 3
50 50	1.0

Problem: Fastestest Function: Observations

- We receive the result of the following equations:

$$x = \frac{a_x}{b+a_x} \text{ and } y = \frac{a_y}{b+a_y}$$

where a_x is the time spent on foo for x and b is the remaining runtime of the program.

- The factor we are looking for is calculated by $\frac{a_x}{a_y}$.

- Rewrite the two equations to a_x and b_x :

$$x = \frac{a_x}{b+a_x} \equiv bx + a_x x = a_x \equiv bx = a_x - a_x x \equiv bx = a_x(1-x) \equiv a_x = \frac{bx}{1-x}$$

Resulting in $a_x = \frac{bx}{1-x}$ and $a_y = \frac{by}{1-y}$.

- filling the factor formula:

$$\frac{a_x}{a_y} = a_x a_y^{-1} = \frac{bx}{1-x} \cdot \frac{1-y}{by} = \frac{bx(1-y)}{(1-x)by} \equiv \frac{x(1-y)}{y(1-x)}$$

- Calculate the factor by the formula, resulting in $\mathcal{O}(1)$ solution.

Solution in C++

```
1  #include <iostream>
2  using namespace std;
3
4  signed main() {
5      long double x, y;
6      cout << setprecision(20);
7      cin >> x >> y;
8      cout << (1/(1-x/100)-1)/(1/(1-y/100)-1) << endl;
9      return 0;
10 }
```

Solution in Java

```
1 import java.util.*;
2 import java.io.*;
3
4 public class DAPCF {
5     public static void main(String[] args) throws IOException {
6         Scanner scanner = new Scanner(System.in);
7         int x = scanner.nextInt();
8         int y = scanner.nextInt();
9         double ans = x / (((1.0 * (100 - x) / (100 - y)) * 100.0) - (100 - x));
10        System.out.println(ans);
11    }
12 }
```

Solution in Kotlin and Python

```
1 fun main() {  
2     val (x, y) = readln().split(" ").take(2).map { it.toDouble() / 100.0 }  
3     println((x * (1 - y)) / (y * (1 - x)))  
4 }
```

```
1 import sys  
2  
3 x, y = [int(x) / 100 for x in sys.stdin.readline().split()]  
4 print((x * (1 - y)) / (y * (1 - x)))
```

Practising between sessions

- All problems from DAPC 2022 and BAPC 2022 are available at <https://domjudge.ewi.tudelft.nl/>, self-register a team.
 - Next three sessions have their own contest
 - All sessions contain similar-themed problems
-
- | | |
|-----------|---|
| Session 2 | Ad-hoc and Math solutions |
| Session 3 | Sort and Search |
| Session 4 | Interactive Problems, Dynamic programming, Divide and Conquer |
-

Meet and Greet

Looking for team?

If you are looking for a team, please raise your hand. If you want, you can give an introduction in the front, like experience and programming languages known. Please don't forget to register at wisv.ch/dapc.

Next session is on <Insert date and location>.

<https://domjudge.ewi.tudelft.nl/>