(While I take attendance)
Make a kattis account if you haven’t yet

open.kattis.com

Getting started with competitions

Greg (Grigorios) Mazonakis (in charge of labs (what is a lab))

Today's agenda:

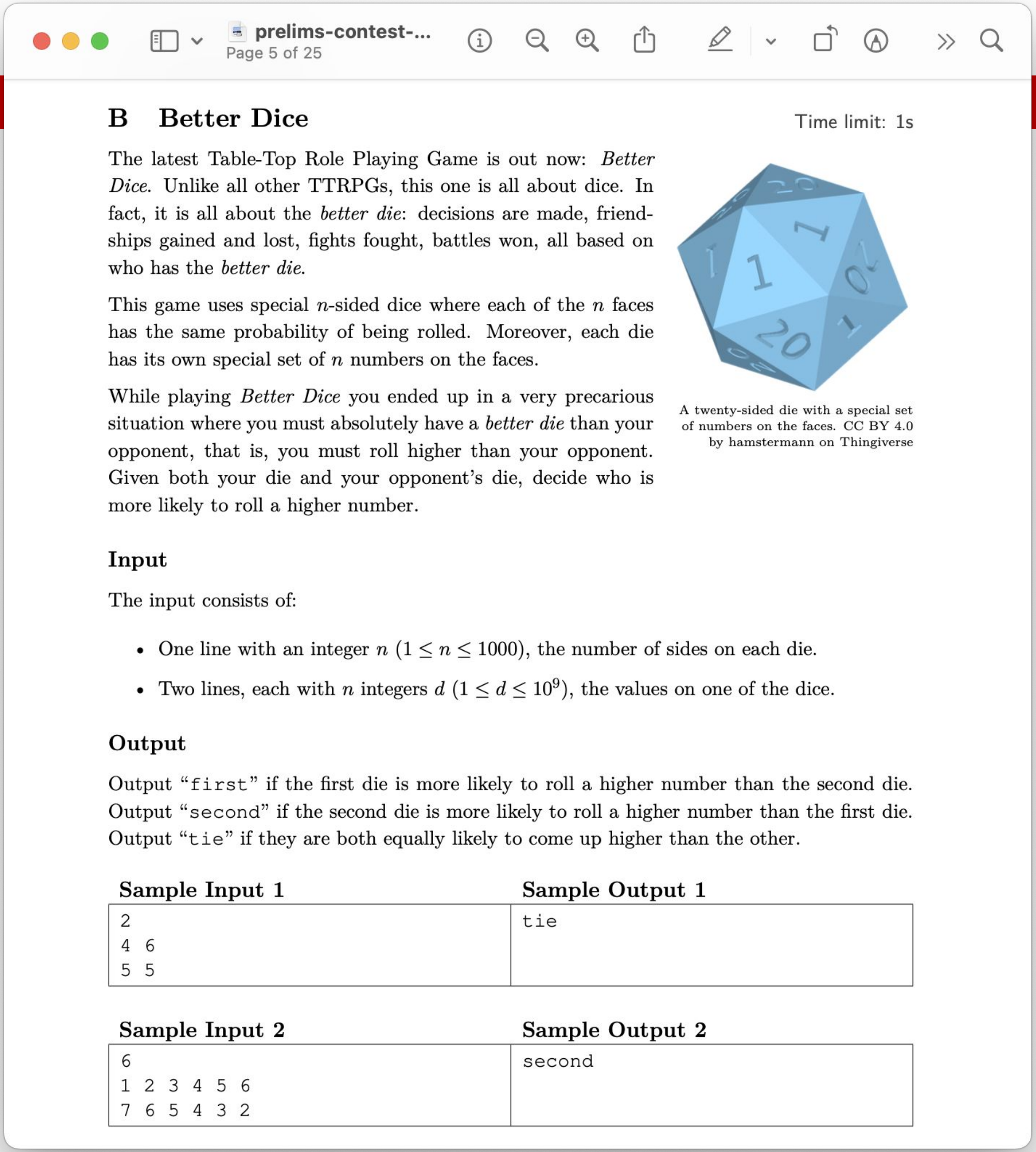
- Short presentation (ask!), skip if already known
- 10-min break
- Team Contest (mini (40 minutes))
- Results & Presentation of solutions

Nature of contests

- Teams of up to 3, only one computer
- 5 Hours, ~12 Problems
- Every person gets a booklet of all problems
- Scoreboard / freeze
- Ranking system
- You can print!
- TCR!!! DOCS!!! NO INTERNET!!!

Nature of problems

- A problem description with quite well/formally specified input/output
- Test cases that assert input-output pairs
  - A few visible sample test cases
  - Lots of hidden test cases
  - No partial points



B Better Dice

Time limit: 1s

The latest Table-Top Role Playing Game is out now: *Better Dice*. Unlike all other TTRPGs, this one is all about dice. In fact, it is all about the *better die*: decisions are made, friendships gained and lost, fights fought, battles won, all based on who has the *better die*.

This game uses special n-sided dice where each of the n faces has the same probability of being rolled. Moreover, each die has its own special set of n numbers on the faces.

While playing *Better Dice* you ended up in a very precarious situation where you must absolutely have a *better die* than your opponent, that is, you must roll higher than your opponent. Given both your die and your opponent's die, decide who is more likely to roll a higher number.

A twenty-sided die with a special set of numbers on the faces. CC BY 4.0 by hamstermann on Thingiverse

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 1000$), the number of sides on each die.
- Two lines, each with n integers d ($1 \leq d \leq 10^9$), the values on one of the dice.

Output

Output "`first`" if the first die is more likely to roll a higher number than the second die. Output "`second`" if the second die is more likely to roll a higher number than the first die. Output "`tie`" if they are both equally likely to come up higher than the other.

Sample Input 1	Sample Output 1

2
4 6
5 5
	tie

Sample Input 2	Sample Output 2

6
1 2 3 4 5 6
7 6 5 4 3 2
	second

Different kinds of problems

- Simulation problems:
  Play out a scenario correctly. Usually easy

- Math problems:
  Find a formula that calculates the solution

- Typical:

  Find the right algorithm for the job, or modify/combine known algorithms. Graph problems are almost always like this

& More…

Form of test cases

single case

```
Input
4 (No of numbers)
8 7 1 3

Output
1 3 7 8
```

multi case

```
Input
2 (No of test cases)
4 (No of numbers)
8 7 1 3
3 (No of numbers)
9 4 5
Output
1 3 7 8 (Case 1)
4 5 9 (Case 2)
```

Interactive

```
In: 8
Out: 8
In: 7
Out: 7 8
In: 1
Out: 1 7 8
In: 3
Out: 1 3 7 8
In: END
exit(0)
```

Judging software

Kattis / Domjudge etc.

- You upload your source code, it compiles it
- It attempts input-output pairs (test cases) sequentially until it finds one that does not pass
- A Pass on all test cases means you passed the problem

Possible judgements:

- Pass
- Wrong answer
- Time limit exceeded
- Error (exit code != 0)
- Failed to compile (what have you done)

Macro-strategy

- Everyone must see everything (ideally)
- Get easy problems out of the way as soon as possible
- Remember what you’re trying to maximize
- Experiment a lot and find out what works best *for your team*
- Pair programming works well for some teams but is very costly
- Take advantage of printing (working in parallel is great)
- Take advantage of what information the scoreboard gives you

Micro-strategy

- Does binary search work here?
- What category of problem is this?
- What are the limits on input size? Reason about time complexity
- Test before you submit, if easy, generate a worst-case input (e.g. Largest prime number smaller than 10^9 should be easy to generate and for some problem can tell you whether your program will time out)
- Be VERY careful about your pipeline (accidentally testing old executable, compilation failing and you not noticing)
- Read very carefully, a single word can change the problem vastly
- Try the test cases yourself
- Problems are sometimes much simpler than they look
- No specialized knowledge required

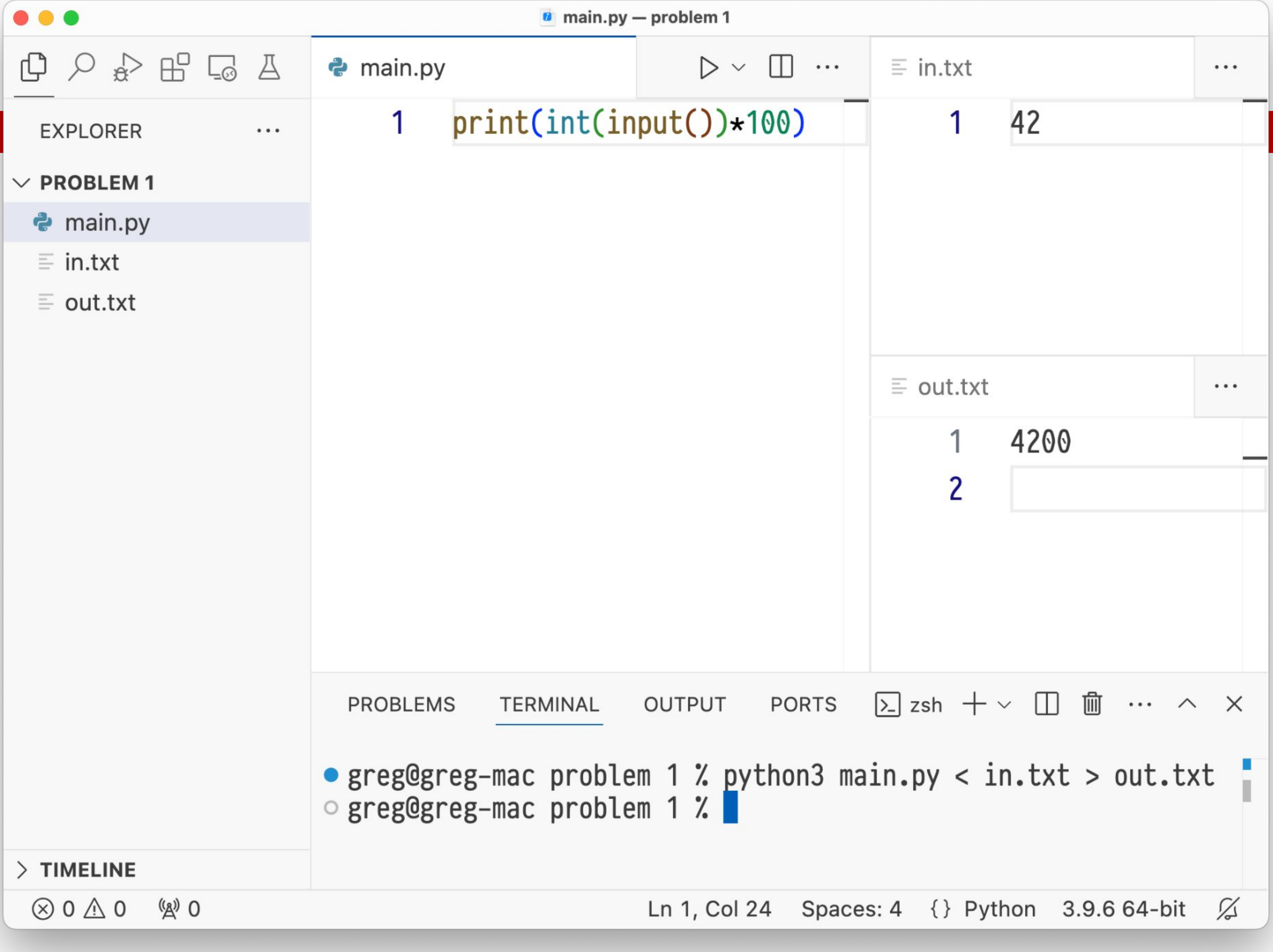
rijksuniversiteit groningen

Workflow

Find something that facilitates quick testing, especially against the provided test cases.

```
exe < in.txt > out.txt
```

Find what works best for your team



main.py:

```
print(int(input())*100)
```

in.txt:

```
42
```

out.txt:

```
4200

```

```
greg@greg-mac problem 1 % python3 main.py < in.txt > out.txt
greg@greg-mac problem 1 %
```

Choice of programming language

- In our contests we allow you to submit C++, Python3, and Java
- Certain problems are sometimes more well-suited to different programming languages (python handles big integers out of the box).
- Use whatever you are familiar and comfortable with. Keep in mind what your team can use and read and work on.
- As with everything, try to figure out what works best for you and your team

Don’t feel lost!

- Lots of resources
- Lots of TAs
- Lots of room to practice

Questions → Break→ Contest!

Send your Kattis email in #contests-questions before the break starts