Armenian Open Programming Competition

In memory of Vladimir Yeghiazaryan

Solutions

EIF ՁԵՌՆԱՐԿՈՒԹՅՈՒՆՆԵՐԻ ԻՆԿՈՒԲԱՏՈՐ ՀԻՄՆԱԴՐԱՄ ENTERPRISE INCUBATOR FOUNDATION

ONEMARKETDATA

РОССИЙСКО-АРМЯНСКИЙ УНИВЕРСИТЕТ

Problem A: Santising

```
print(x * 0.6)
```

Problem B: Hidden text

One simple solution

- Copy the file into a text editor
- Replace all tabs to nothing
- Manually find the answer

hayastan

Problem C: Drinking party

- This is the famous NIM problem

```
if n % 3 == 2:
    take 1 glass
else:
    take 2 glasses
```

- Easy to deduce by trying to play the game for n = 1...10

Problem D: Look-and-Say numbers

- Simply simulate the answer
- For the first 15 items see: https://oeis.org/A005150

Problem E: Roman roads

- Simulate the process starting from the car that left first
  - The first car never gets stuck and finishes as if it were alone
- For every car, it finishes either
  - At the “normal” time without being stuck; or
  - Gets stuck and finishes with a car…
    - that finishes latest among those that left earlier and are wider!

Key observation!

whichever is later

- Can be implemented using a segment tree or a binary indexed tree.

Problem F: Formula One

- Use API from http://ergast.com/
- Scrape some website e.g. https://www.statsf1.com/
- Ergast and Kaggle provide offline database, which you can download and write a SQL query to get the required data.
- Wikipedia(?)
- ...

We did the first two and the results matched.
Ergast provides Rest API of a form http://ergast.com/api/f1/${year}/${round}/results

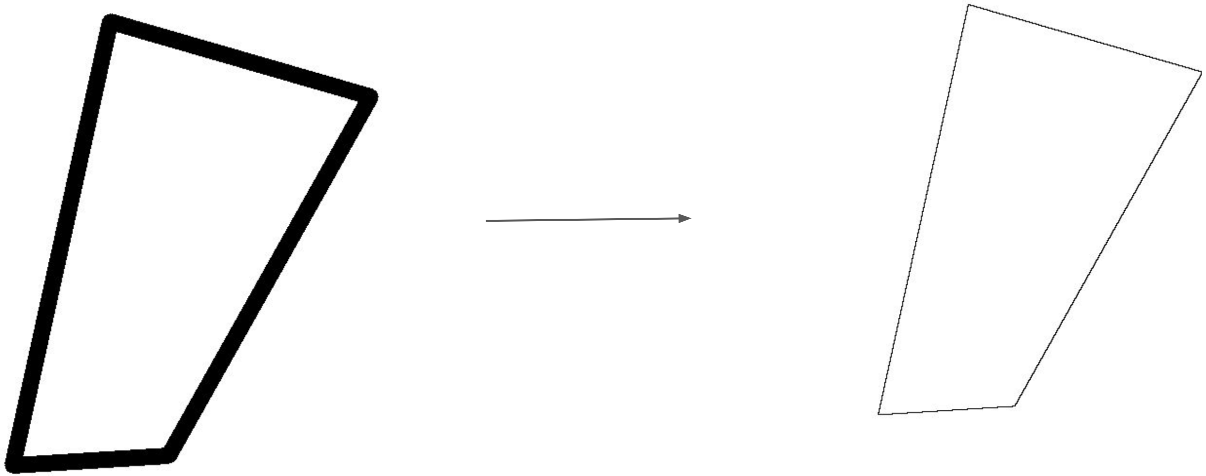
Problem G: Can you unzip me?

- Unzipping twice will crash something on your computer.
- Unzipping once is ok, so we do that
- Second step of unzipping and filtering whitespace characters we do in one step

  ```
  unzip -p big.zip | grep -o '\S'
  ```
- Runs about 30 minutes.

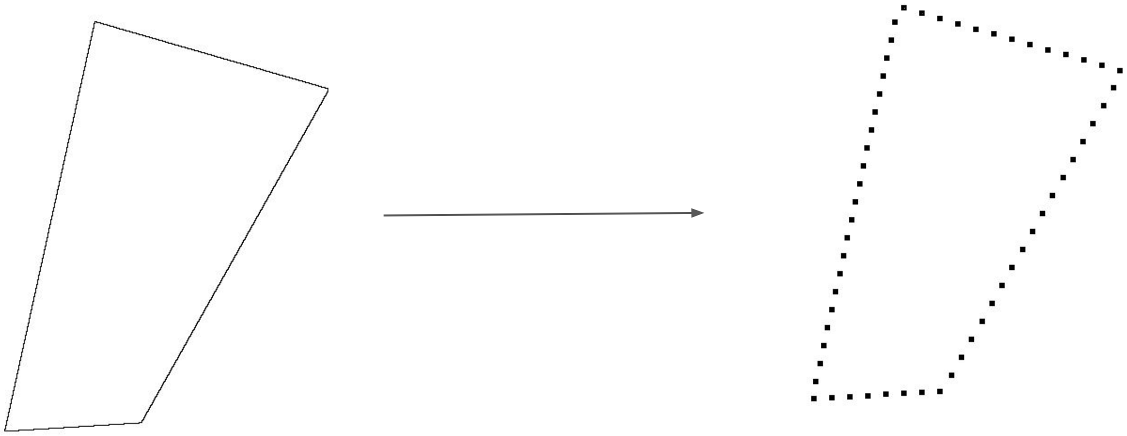
Problem H: Virus shapes (a possible solution)

- Thin the shell using flood fill inside



Problem H: Virus shapes (a possible solution)

- Use BFS to track every (say) 20th point along the contour



Problem H: Virus shapes (a possible solution)

- Compare the angles of consecutive triplets of points, they should all be about 180 degrees, except when there is a corner.

- Knowledge of opencv, python and numpy helps in some parts of this problem, but are not strictly necessary.

Problem I: Earthovirus

- Randomly check if substrings of length 25 of E are in the parts of X that are remaining.
- If at least one of them is there, then it is an Earthovirus
  - We know this because X is random
  - If X were adversarially chosen, this algorithm could not work.

Problem J: Statistics

- Another relatively standard data structure problem
- Can be solved in O(NlogN) using std::set or in O(N) using a stack.

Problem K: Virus modelling

- Key observation: relativity
  - Can assume that Alice is at the origin and is not moving
- The problem is reduced to an intersection of a *filled* circle with a segment
  - Find an intersection of a line and a circle
  - Cover the corner cases e.g. Bob being inside the circle and never leaving.
- See e-maxx for algorithms on intersection of line and a circle

Problem L: Contest

- Key observation 1
  - Hayk should always propose the problem with the largest q_i* s_i value, where q_i is the updated probability for problem i.
- Key observation 2
  - Consider some set S of problems that have not yet been accepted.
  - If the total number of attempts Hayk did on problems in S is m, then we can calculate exactly the updated probabilities q_i for i in S by simulating m steps of the process using KO1.
- Using KO1 and KO2, we can solve the problem using DP.
- The state of the DP is

  (set S of solved problems, total # of attempts, total # of attempts on S)
- Size of the state space is O(N^2 * 2^T), which is completely manageable.

Problem M: Buildings

- Create a bipartite graph with X coordinates on the left side and Y coordinates on the right side.
- A path of length three has to close to a cycle of length 4
- Closing all cycles of length 4 means transforming a bipartite graph to a full bipartite graph.
- Use BFS to find all connected components of the bipartite graph
- Answer will be $R_1 * C_1 + R_2 * C_2 + \ldots$ where R_i, C_i are number of upper/lower vertex of i-th component.

Problem N: Lockdown

- Given some time t, the positions of all officers can be determined.
- Thus, can do DP with the state space (time, position of Ashot).
    - There is a problem: We do not know how large time can be.
    - In fact, it can be so large that this approach does not work.
- Improved version:
    - The state of the officers’ positions repeats every 120 steps (at most), because we may have cycles of even length only (e.g. 1 2 3 2 or 1 2 3 4 5 4 3 2) and they will repeat in LCM(2, 4, 6, 8, 12) = 120.
    - So the **time dimension is not unbounded.**
- Can be solved using Dijkstra or DP over the space of size 120 x R x C