TWGSS
OI Team

2024/25 HKOI Final Mock II

Editorial

6 December 2024

Statistics

	Problem	Attempts	Full	Max	Mean	Std Dev	LQ	Median	UQ	Subtasks
TMF255	MINESWEEPER 1D	14	100	100	21.714	29.104	0	7.5	39	6 | 7, 9 | 6, 17 | 4, 24 | 4, 15 | 1, 29 | 1
TMF256	Treasure Detector	9	100	100	43.111	42.694	3.5	19	100	7 | 6, 11 | 5, 12 | 5, 21 | 4, 33 | 3, 16 | 3
TMF257	TOWER OF GOD MAGIC	7	100	100	16.429	34.508	0	0	15	5 | 1, 15 | 2, 20 | 1, 60 | 1
TMF258	BEYOND 2048	12	100	100	15	26.555	0	9	9	9 | 8, 17 | 2, 23 | 1, 22 | 1, 29 | 1
		16	**400**	**400**	**61.688**	**102.686**	**9**	**17**	**58**	

Prizes Estimation

This time, the judgement will base on HKOI Senior (harsher!)

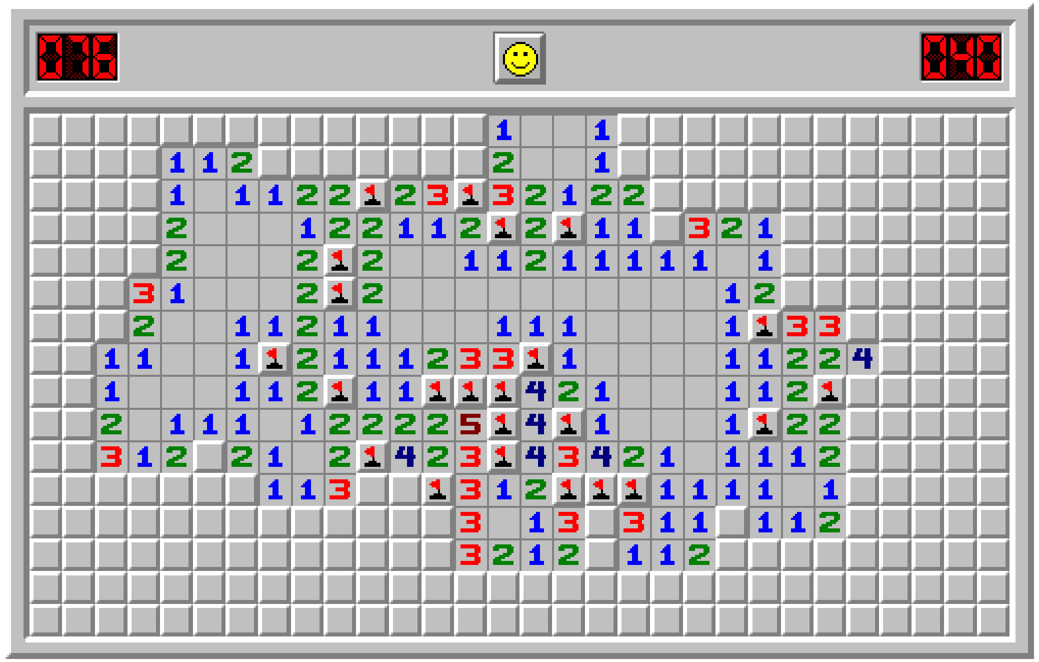
			Gold	Silver	Bronze	HM
Expected Score	*TMF255*	Minesweeper 1D	100	100	71	32
	TMF256	Treasure Detector	100	100	30	18
	TMF257	Tower of God Magic	100	64	28	12
	TMF258	Beyond 2048	71	49	26	9
	Total		371	313	155	71
Multiplier			×0.925			
Cut-off			**343**	**290**	**143**	**66**
No. of Recipients			1	-	2	-

MINESWEEPER 1D

TMF255

Problem Statement

- Given a **1D** minesweeper board, determine if it is possibly legit.
- If it is possibly legit, construct a possible bomb configuration.



Subtask 1

S does not contain any `?`

IQ Test!

- What if S does not contain any `?`?
- The game is completed!

- Therefore, if S only contains `0`, it is `Possible` and output `.....`... (N dots).
- Otherwise, output `Impossible`.

Expected Score	6

Cumulative	6

Subtask 2

S contains at most one `?`

It is guaranteed that solution exists

Some implementation idea:

- Find the `?` (if exists).
- If its neighbour is `1`, then it is a bomb.
- Otherwise, it is safe.

Expected Score	9

Cumulative	15

Subtask 3

Two digits in S are separated by at least two `?`

- The key idea is to put bombs according to the digits.
- The bomb put won't interfere others as the gap of at least two ?
- In such situation, solution always exists? No!
  - Consider `2` at the side of the board!
- A way to resolve this in general is to check whether your construction matches the information given.

Expected Score	17

Cumulative	32

Subtask 4

S does not contain any `1`

- When there is no `1`, the adjacent cells of the digits are actually fixed to be both bombs or both safe.
- Therefore, you can simply place the bombs according to the hints.
- Also, you shall check whether or not your construction is correct.
  - e.g. The board `?02?` obviously has no solution.

Expected Score	24	Cumulative	56

Subtask 5

$1 \leq N \leq 5000$

It is guaranteed that solution exists

- After handling all the `0` and `2`s, what remain to handle is all the `1`s.
- Key issue with `1`s is that there are two possible ways to handle the `1`s:
  - Left bomb, right safe.
  - Left safe, right bomb.
- Therefore, we can repeatedly "scan" the board:
  - If either neighbour of an `1` is decided (including board boundary), we can decide the other neighbour.
  - If none of either side of all the `1` is decided, we can simply alternate `.#.#.#.#` and it will works! (Why?)

O(	N^2	)	Expected Score	**71**	Cumulative	71

Full Solution

- We may handle the `1`s more efficiently by using a vector of to handle 1s:
  - After handle each `1`, we push the "close" element of it to the vector if there is change in its neighbour.
  - For each `?`, it is handled at most once and so the total number of operations is O(N).

- Indeed, another solution is to use dynamic programming – you can investigate that if you are interested.

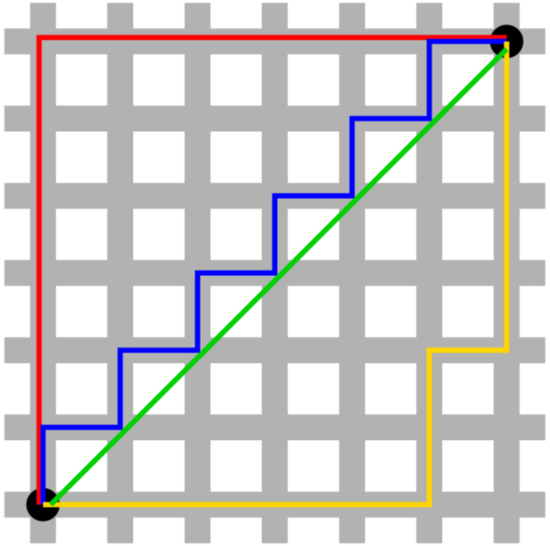
O(	**N**	)	Expected Score	**100**	Cumulative	100

Treasure Detector

TMF256

Problem Statement

- Given a board of $N \times M$ $(N, M \leq 10^9)$ cells.
- Luna pick a cell, you need to pick **another** cell.
- Pick any cell such that most number of cells is closer to your cell when compared to Luna's.
- Distance: Manhattan $|x_1 - x_2| + |y_1 - y_2|$



Subtask 1

$N = 1$
$1 \le M \le 10^6$

- It is a just a 1D number guessing game, whoever closer wins.
- What is the optimal strategy?
  - Next to the opponent's cell!
- Therefore, you can simply check both possibility (left and right), and find the better one.

O(	M	)	Expected Score	7	Cumulative	7

TWGSS OI Team

Subtask 2

$2 \leq N \times M \leq 1000$

- With such small constraints, we can simply brute force check all the $N \times M - 1$ possible choices in O(NM) time for each.

O(	**(NM)²**	)

Expected Score	**11**

Cumulative	18

Subtask 3

$2 \leq N \times M \leq 10^6$
$R = C = 1$

- Luna picks the top left cell.
- How to respond?
  - Using the same idea in Subtask 1 – it must be either $(1, 2)$, $(2, 2)$ or $(2, 1)$!
  - We may simply check all of the possibilities.
  - Make sure do not check out-of-bound cells.

O(	**NM**	)	Expected Score	**12**	Cumulative	30

Subtask 4

$2 \leq N \times M \leq 10^6$

	(Y)	
(Y)	L	(Y)
	(Y)	

- In fact, the observation in the 1D version also works here!
- If Luna's choice is (R, C), then the optimal respond must be one of the followings: $(R+1, C)$ $(R-1, C)$ $(R, C+1)$ $(R, C-1)$
- Why? The logic behind is if otherwise, we can always move "closer" such that the number of cells winning is unchanged or increased.

- Therefore, we can check the each of the four possibilities in O(NM) time.

O(	**NM**	)	Expected Score	**51**	Cumulative 51

TWGSS OI Team

Subtask 5

$1 \leq N, M \leq 10^6$

- The bottleneck is very clear: how do we compute the number of winning cells quickly?
- When $N, M \leq 10^6$, a choice is to notice that the winning cells in each row must be forming a range of either from 1 to X, or from Y to M.
- Therefore, for choosing the left / right cell, we can somehow loop from 1 to X, or from Y to M to count the number of columns you win, then multiply it by the total number of rows.
- It's the same idea for choosing the top / bottom cell.

O(**max(N, M)**)	Expected Score	**84**	Cumulative	84

Full Solution

"Therefore, for choosing the left / right cell, we can somehow loop from 1 to X, or from Y to M to count the number of columns you win, then multiply it by the total number of rows."

- Is this serious?
- Instead of using loop, we can just calculate the number of rows / columns it easily, which is X or $M - Y + 1$...

- Subtask 5 is somehow useless.

O(	1	)	Expected Score	100	Cumulative	100

TOWER OF GOD MAGIC

TMF257

Problem Statement

- Given N kinds of gems, each of three.
- The gems are put onto a $3 \times N$ board.

- Each move consists of the followings:
  - Put down your finger.
  - Move repeatedly to (corner-)adjacent cell and swap with the gem in that cell.

- Target: same kind of gem in the same column while using **the minimum number of moves**.

I wonder what is this game?

Partial Scoring

- K: the number of moves
- The author is very kind in giving partial scoring, as the original idea is must use not more than one move (always possible)!

SUBTASKS

	Points	Constraints
1	5	The game can be finished in at most one swap
2	15	$N = 2$
3	20	$2 \le N \le 3$
4	60	No additional constraints

- If $K \le 1$, then you score 100%.
- Otherwise, if $K \le N$, you score 60%.
- Otherwise, if $K \le 2N$, you score 50%.
- Otherwise, if $K \le 3N$, you score 40%.
- Otherwise, if $K \le 10^5$, you score 20%.
- Otherwise, you score 0.

Subtask 1

The game can be finished in at most one swap

- Find the “incorrect” gems.
- Swap them.
- Make sure you understand the output format!

- Note that the given board may already be solved, so no move is needed!
- Your code have to check this probability too.

Expected Score	5

Subtask 2 & 3

$2 \leq N \leq 3$

- Some small-sized board for you to (slightly) test your solution.
- Some very random solution ideas for these two subtasks:
  - Hand-solved the cases and hardcode them (not many cases!)
  - Randomly move to adjacent cells repeatedly until correct.
  - Write the full solution (will talk later!)

Expected Score	**3 - 35**

(depends how good you do in partial score / subtask)

Method 1: Only Swapping

$K \leq 10^5$ (20%)

- If each move consists of a swap only, it is much simpler!
- There is a very simple idea:
  - Target: $1, 2, \cdots, N$ in each row.
  - Construct each column one-by-one. How?
  - Find a gem that is the kind of the column number and swap it with a gem (of different kind) on the left side of it (need to special handle if this is not possible)

Expected Score	20

Cumulative	52

Method 1+: One move per gem

$K \leq 3N$ (40%)

- Actually we are always swapping the same gem when we try to put it to a certain location.
- Therefore, we can hold it for longer until it arrived at the destinated location!

Expected Score	**40**

Cumulative	64

Method 1++: One move per gem PRO

$K \leq 2N$ (50%)

- As it is not necessarily to put the gems in order of $1, 2, \cdots, N$, we can instead decide what to put in that column as the current gem at the top cell.
- This reduces the number of moves to $2N$.

Expected Score	**50**

Cumulative	70

Method 2: One Move Only

- In fact, we can chain the swaps using some “interesting” ideas!

- I believe that the idea is easy as a game – the hard part of this problem is probably its implementation.
- The key idea is to solve the board columns by columns and reduce it to a smaller board.
- A good idea is to simulate your answer step-by-step, even if no visualisation is provided!

Expected Score	100

Cumulative	100

Side Notes

- It is a constructive task – so there can be other ideas to solve this problem.
- Sometimes a good idea to simulate your answer using program (instead of thinking using hand!)

BEYOND 2048

TMF258

Problem Statement

- Given a row of N tiles, each with a number written (power of 2)
- Each move, you can
  - merge two adjacent identical tiles, and
  - obtain a score equals to the number on the spawned tile.
- Your target is to maximise the score.

```
5
8 2 2 4 16
```

Subtask 1

$A_1 = A_2 = \cdots = A_N = 2$

- Clearly, when all cells are 2, we can simply "merge as we want".
  - Whenever there are two cells with identical numbers, we can merge them.

- You may do the math, or simply simulate the process to get the final score of the game.

O(	N	)	Expected Score	9	Cumulative	9

Subtask 2

$1 \leq N \leq 5000$
$A_1 \leq A_2 \leq \cdots \leq A_N$

- The number on the tiles are sorted.
- Is there anything special about that?
- Turns out yes – the previous claim still works!
  - Whenever there are two cells with identical numbers, we can merge them.
- Because, we can always keep the same number tiles "together".
  - By always merging the rightmost two tiles among same number tiles.

- Therefore, you may again do some math / simulate this process

O(	N N²	)	Expected Score	17 26	Cumulative	26

Subtask 3 & 4

$1 \leq N \leq 300 \mathbin{/} 5000$

- Okay, the actual problem starts here.
- In fact, greedy algorithms do not work well in the general case.
  - It is hard to decide how to merge!
- Consider Sample 3:
  - Merge left (4 4 2 4 8 → 8 2 4 8)
  - Merge right (4 2 4 4 8 → 4 2 8 8 → 4 2 16)
  - Merge right better!

```
6
4 2 2 2 4 8
```

Subtask 3 ~ 4

$1 \leq N \leq 300\ /\ 5000$

- Start from this subtask, dynamic programming is needed (probably).
- Suppose we have the value of
  - `dp[R]`: max score obtained from tiles $1 \cdots R$
  - `score[L][R]`: score of merging all the tiles $L \cdots R$ to **one** tile (-infinity if impossible)

- Then the answer is just `dp[N]` (obviously)!
- But how can we compute the values?

Subtask 3 ~ 4

$1 \leq N \leq 300 \ / \ 5000$

- We loop `L` from `0` to `R-1`,

  `dp[R]` = maximum value of `dp[L] + score[L+1][R]`
- `score[L][R] = score[L][M] + score[M+1][R]` if we can find a `M` such that the number on merged tile by tiles `L`...`M` and tiles `M+1`...`R` are equal (always unique), and −infinity otherwise.
- One may optimise by using partial sum / binary search to find `M`.

O(		)	Expected Score		Cumulative	
	N^3			**23**		49
	N^2 lgN			**62**		71
	N^2			**62**		71

Full Solution

- Full solution is actually an extension of Subtask 3 ~ 4.
- Note the largest tile can be merged is $N \times 2048$, which is about 2^{30}.
- So there are at most 30 values of `score[L][R]` that is not –infinity!
- Therefore, you may instead compute only those right bounds, and use the same main idea of range DP.

O(	**N lgN** **N lg^{2}N**	)	Expected Score	**100**	Cumulative	100

TWGSS OI Team