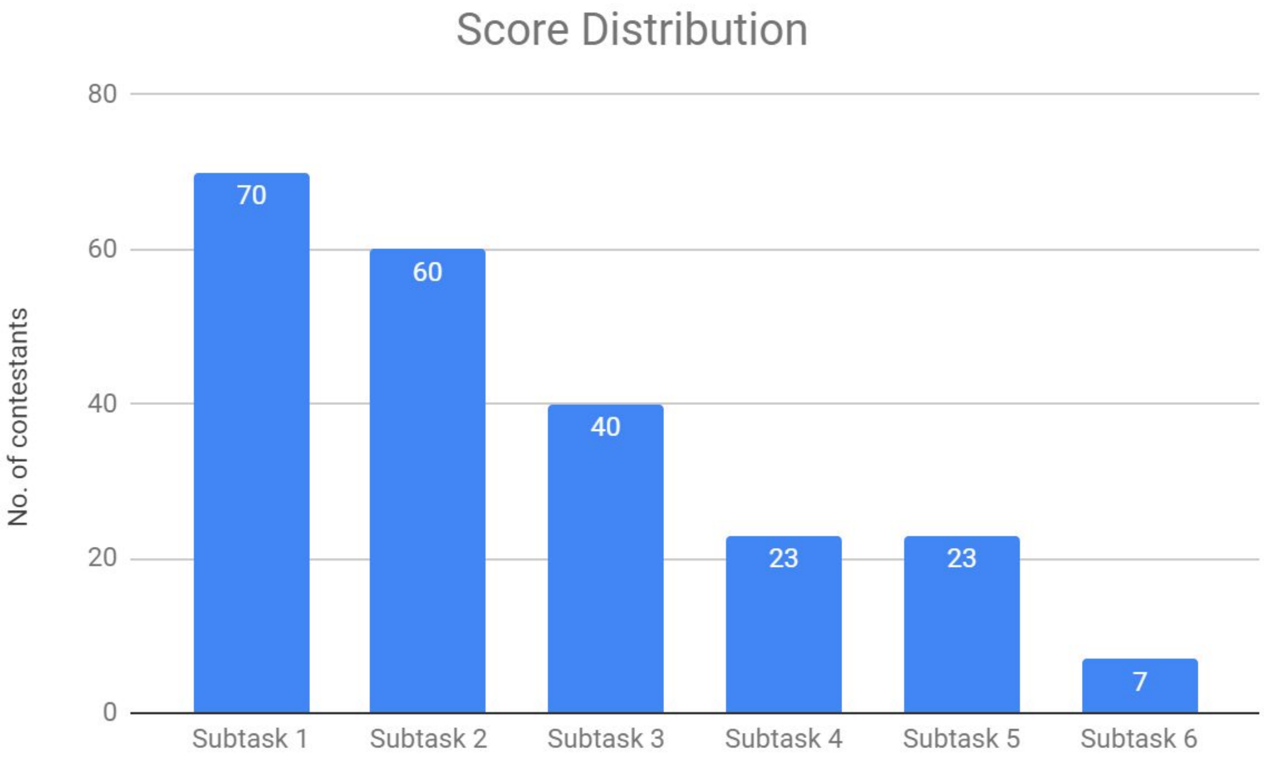
Snakes and Snakes

Author : Alex Tung
Speaker : Jeremy Chow
26/1/2019

Statistics


Score Distribution
80
60
40
20
0
No. of contestants
70
60
40
23
23
7
Subtask 1
Subtask 2
Subtask 3
Subtask 4
Subtask 5
Subtask 6

Statistics

- Attempts : 78
- Max : 100
- Mean : 28.435
- Std Dev : 29.331

- First Solved by Jamie Choi at 0:25
- Highest mean among senior problems

Problem Statment

- Given a board consists of N+1 squares and M “Snakes”
- Each time you can move forward 1 - K squares
- Find the minimum number of moves needed to move from square 0 to square N
- And determine whether you can do it

SUBTASKS

For all cases:
$1 \le K \le N \le 10^9$
$0 \le M \le \min(N-1, 200000)$

	Points	Constraints
1	3	$M = 0$
2	4	$K = 1$
3	13	$K = 2$ $2 \le N \le 200000$
4	24	$K = 2$
5	19	$K \le 100$ $N \le 200000$
6	37	No additional constraints

Subtask 1

- M = 0
- No snakes on the board
- Just calcuate the answer by math
- Ans = ceil(n / k) = (n - 1) / k + 1

- Time Complexity = O(1)
- Easy 3 points :)

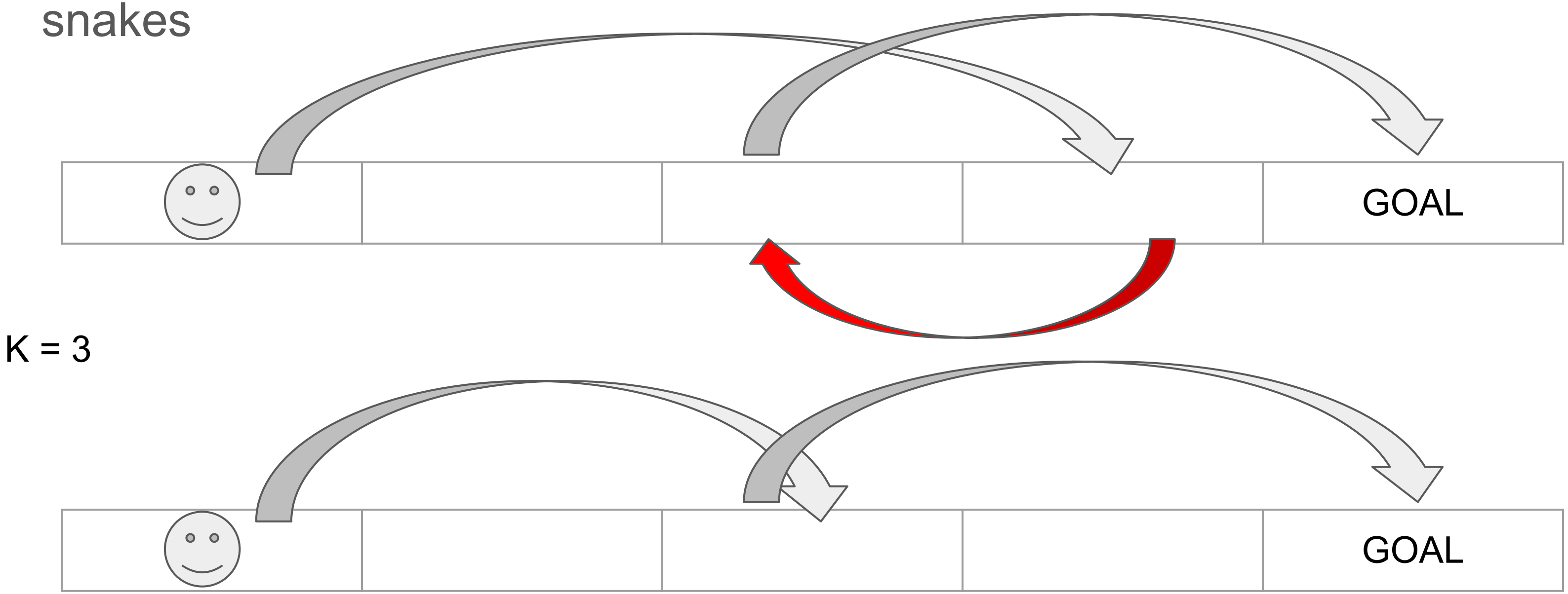
Subtask 2

- K = 1
- Each time you can only move 1 square forward
- If there is any snake, which will bring you backward
- It is impossble to go to square N
- Else you can always move to square N by N moves

- Time Complexity = O(1)
- Easy 4 points :)

Subtask 3

- Observation : There always exist an optimal solution which won't use the snakes



Subtask 3

- The problem become….
- Given a board consist of N+1 squares and M of them are forbidden
- Each time you can move forward 1 - K squares
- Find the minimum number of moves required to move from square 0 to square N / determine whether you can do it

Subtask 3

- K = 2, 2 <= N <= 200000
- N is small
- Just simulate the process
- Notice that it is optimal to go to the farthest square which is not forbidden

Subtask 3

- Let you are now at square i
- If square i+2 is not forbidden, go there
- else if square i+1 is not forbidden, go there
- else return impossible
- Repeat this process until you get to square N

- Time complexity = O(N)

Subtask 4

- K = 2, N could be very large, up to 10^9
- We can use the fast forward technique
- For each snake, we calculate the number of moves needed to get over it

- Let you are at square x and the snake entry is at square y
- if the parity of x and y is not same, then you can ignore that snake
- if the parity of x and y is same, then you need to do some simple calculations

Subtask 4

- After that, update your position
- After you pass over all M snake, you can calculate the amount of move to go to square N just like what we did in subtask 1
- You can check the impossible case by checking whether there exist 2 consecutive snake

- Time complexity = O(M)

Subtask 5

- K <= 100, 2 <= N <= 200000
- K can vary now
- But it is not a big deal

- We can use our algorithm in subtask 3
- check K squares instead of 2 squares

Subtask 5

- Let you are now at square i
- If square i+k is not forbidden, go there
- else if square i+k-1 is not forbidden, go there
- else if ……..
- else return impossible
- Repeat this process until you get to square N

- Time complexity = O(N)

Subtask 5

- Why time complexity = O(N) instead of O(NK) ?
- although both is fast enough to pass this subtask

- In every 2 moves, you can always move K or more squares forward
- else it is impossible
- O(K) * O(N / K) = O(N)

- You can also use a O(NK) dynamic programming to pass this subtask

Subtask 6

- $1 <= K <= N <= 10^9$, $0 <= M <= min(N-1, 200000)$
- N could be very large

- Just like what we did before
- We can use the fast forward technique to solve this problem

Subtask 6

- Firstly, check the impossible case by checking whether there exist K consecutive snakes
- Then, for the first snake which is in front of you, with position y
- move to the square x’ which is one move away from the snake
  - i.e. x’ < y, x’ + K >= y
- After that, move one more move, find the farthest square which is not forbidden and go there
  - can be done by while loop on the snakes
- If you still can’t pass over that snake, i.e. farthest square = y - 1, move one more move

Subtask 6

- Finally, when there is no snake in front of you, calculate the amount of move to go to square N just like what we did in subtask 1
- Remeber to check the impossible case

- Time complexity = O(M)

- Be careful about the calculation

THX