CMIMC 2023

Theoretical Computer Science Round

Instructions

1. Do not look at the test before the proctor starts the round.
2. This test consists of three problems, which require proofs, to be solved within a time frame of 60 minutes. There are 300 points total.
3. No computational aids other than pencil/pen are permitted.
4. Answers should be written and clearly labeled on sheets of blank paper. Each numbered problem should be *on its own sheet*. If you have multiple pages, number them as well (e.g. 1/3, 2/3).
5. Write your team name on the upper-right corner and the problem and page number of the problem whose solution you are writing on the upper-left corner on each page you submit. Papers missing these will not be graded. Problems with more than one submission will not be graded.
6. Write legibly. Illegible handwriting will not be graded.
7. If you believe that the test contains an error, submit your protest to the 2023 CMIMC discord.

CMIMC 2023

Carnegie Corporation Promotion

1. Carnegie Corporation is trying to promote a new department director from its employees. Carnegie Corporation has n employees each with some unknown real-valued score and wants to pick the M-th highest scoring employee to be the new department director. The corporation has a magical machine that, once a day, can be used to compare two employees to see which one has a higher score. Unfortunately, this machine has a magical consequence: after every k uses of this machine, if a new department director has not been chosen by the end of the day, one random employee is fired and a new employee (who has not necessarily the same score) is hired. Assume no two employees have equal scores and scores don't change over time.

   Machines with a higher constant of k will be more expensive, so management wants to minimize the value of k. Devise an algorithm which uses the minimum possible k guaranteeing that, given any $1 \leq M \leq n$, we can promote the M-th highest scoring employee as the new department director in finitely many days.

Scoring

An algorithm that solves the case for a certain k in terms of $n \geq 2$ and some constant $c \in \mathbb{R}^+$ will be awarded:

- 10 pts for any finite k
- 20 pts for any $k \leq cn\log(n)$ for some constant $c > 0$
- 40 pts for any $k \leq cn$ for some constant $c > 1$
- 70 pts for $k \leq n$
- 85 pts for $k \leq n - \lfloor\sqrt{n/2} - \frac{1}{2}\rfloor + 1$
- 100 pts for $k \leq n - \lfloor\sqrt{n} - \frac{1}{2}\rfloor + 1$

You may only submit one solution per problem. **Please specify which bound you are trying to prove for your submission, or you may not be awarded points for that bound.**

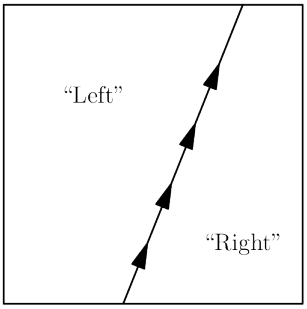
For an incorrect solution, partial points may be awarded for reasonable progress or achieving one of the weaker bounds. Partial points may be deducted for logical flaws or lack of rigor in proofs. To get full points, you must demonstrate that your algorithm always produces a correct result, and always runs in at most the stated number of moves.

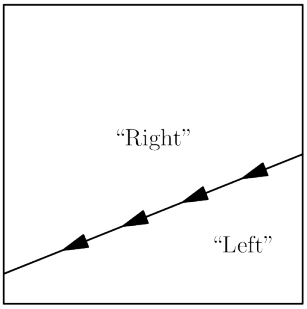
CMIMC 2023

Scottybeard's Treasure

2. After years at sail, you and your crew have found the island that houses the great treasure of Scottybeard, the greatest pirate to ever sail the high seas. The island takes the shape of a unit square, and the treasure (which we treat as a single point) could be buried under any point on the island.

   To assist you in finding his treasure, Scotty has left a peculiar instrument. To use this instrument, you may draw any directed line (possibly one that never hits the island!), and the instrument will tell you whether the treasure lies to the "left" or the "right" of the line.[1]





   However, Scotty also left a trap! If the instrument ever reports "left" three times in a row or "right" three times in a row, the island will suddenly sink into the sea, submerging the treasure forever and drowning you and your crew! You want to avoid this at all costs.

   To minimize the amount of energy spent digging, you would like to narrow down the set of possible locations of the treasure to be as small as possible. However, Scotty left one last trick; you can only use the instrument 12 times before it breaks!

   Devise an algorithm to use the instrument no more than 12 times that can never result in the island sinking and narrows the worst-case space of possible locations of the treasure to have as small an area as possible.

Scoring

An algorithm that achieves a worst-case area of K will be awarded:

- 1 point for any $K < 1$
- 10 points for $K = \frac{1}{4}$
- 20 points for $\frac{1}{128} < K < \frac{1}{4}$
- 30 points for $K = \frac{1}{128}$
- 50 points for $K_{\min} < K < \frac{1}{128}$
- 75 points for $K = K_{\min}$
- 100 points for $K = K_{\min}$, with a proof that this is optimal

(where $K_{\min}$ is the smallest possible worst-case area, which we are not disclosing to avoid giving anything away).

You may only submit one solution per problem. **Please specify which bound you are trying to prove for your submission, or you may not be awarded points for that bound.**

For an incorrect solution, partial points may be awarded for reasonable progress or achieving one of the weaker bounds. Partial points may be deducted for logical flaws or lack of rigor in proofs.To get full points, you must demonstrate that your algorithm never sinks the island, and always achieves at most your claimed worst-case area.

[1] Where "left" or "right" is taken with respect to an observer walking along the line in its designated direction. There is also a probability zero chance the treasure is precisely on the line; this won't affect anything, but for the sake of clarity let's say the instrument reports "left" in this case.

CMIMC 2023

Perfect Shuffle

3. You are given a deck of $n \cdot m$ different cards where n and m are fixed numbers both between 10^{99} and 10^{100}. You perform *m-perfect shuffle* for some times. In a single *m-perfect shuffle*, you divide the deck into m piles with n consecutive cards in each pile. You take one card from each pile, in order of the piles, for n times to form the new deck. (The *m-perfect shuffle* is deterministic)

   For example, if the cards are labeled 12345678 where $n = 4$ and $m = 2$, you divide the deck into 1234 and 5678, and after one 2-*perfect shuffle* you get 15263748. In another example, if the cards are labeled 123456789 where $n = 3$ and $m = 3$, you divide the deck into 123, 456, and 789, and after one 3-*perfect shuffle* you get 147258369.

   Find an algorithm that, in at most k steps, outputs the smallest positive number of *m-perfect shuffle* after which the deck is exactly the same as the original deck. In each step, you can do one arithmetic operation in $\{+, -, *, /, \text{mod}\}$, do one comparison, break out of a loop, or store one number to a specific location of an array. You can use the following precomputed numbers of steps in your solution:

   - Checking if a divides b for any two integers a and b takes 2 steps because you need to compute $b \bmod a$ then compare with 0. - A loop over k iterations takes k steps because you need to increment the loop index by 1 for k times. - Simulating one "m-perfect shuffle" takes $6nm$ steps because there is one loop index increment, four arithmetic operations, and one store in each iteration of the loop.

Scoring

An algorithm that completes in at most k steps will be awarded:

- 1 pt for $k > 10^{10^{10^{10}}}$
- 10 pts for $k = 10^{10^{10^{10}}}$
- 20 pts for $k = 10^{420}$
- 30 pts for $k = 10^{360}$
- 50 pts for $k = 10^{240}$
- 70 pts for $k = 10^{202}$
- 80 pts for $k = 10^{201}$
- 95 pts for $k = 10^{120}$
- 98 pts for $k = 10^{102}$
- 100 pts for $k = 10^{101}$

You may only submit one solution per problem. **Please specify which bound you are trying to prove for your submission, or you may not be awarded points for that bound.**

For an incorrect solution, partial points may be awarded for reasonable progress or achieving one of the weaker bounds. Partial points may be deducted for logical flaws or lack of rigor in proofs. To get full points, you must demonstrate that your algorithm always produces a correct result, and always runs in at most the stated number of steps.