Solution Sketches

CS@Mines High School Programming Competition 2025

April 26, 2025

Colorado School of Mines

Class Photo (Gavin Garzia)

Summary

Calculate how many students are in the photo given that the students form a rectangle that is w people by l people

The solution is to read the two integers and multiply them together to find the total number of people in the photo.

Fair Grading (Adeline Greene)

Summary

Compute the student's **final grade** given the two midterm exams grades x and y, and the final exam grade z.

- Compute the *weighted average* of the three grades:

$$G_{\text{final}} = 0.25 \cdot x + 0.25 \cdot y + 0.5 \cdot z$$

- Then, use a bunch of `if` statements to determine the letter grade.

Certianly (Jayden Pahukula)

Summary

Calculate how many times the string ”certainly” appears in the input string *s*

The solution was searching the input string for the substring ”certainly”. This can be done by using the substring method in your language or by iterating through the string via a loop.

Hill Climb Racing (Nathan George)

Summary

Given an array of $l + 1$ numbers check whether adjacent elements increase by at most a.

Use a loop to read in the array and a second loop to check the condition for each pair of numbers.

Cheese Touch (Adeline Greene)

Summary

Given the time it takes for the cheese touch to spread p and the time it takes to find a cure t, calculate whether the cure is found before all healthy people become infected.

Let s be a single string consisting of the characters '`H`', '`I`', and '`W`' representing the initial configuration of Healthy people, Infected people, and Walls, respectively.

The solution is to simulate the people infected at all times $p, 2p, 3p, \ldots$ less than t.

This means that for each Infected person, any Healthy person directly to the left or right becomes Infected.

If there are no Healthy people left at any time less than t, print "`ALL INFECTED`". Otherwise print "`CURED`".

Golden Landmarks (Gavin Garzia)

Summary

Given a list of landmarks, their locations, and a specific order in which to visit the landmarks, find the total walking distance required to visit all landmarks.

For each of the n landmarks, store the landmark name s and the landmark coordinates x and y. This can be easily done with a dictionary.

Then, loop through the sequence of landmark names, calculating the distance $|x_i - x_{i+i}| + |y_i - y_{i+i}|$ between each landmark i and $i + 1$ for $0 \leq i \leq n - 2$. Add all these distances to a total distance count.

Club Pizza (Jayden Pahukula)

Summary

Given a list of clubs' meeting times and pizza slices offered, help Blaster maximize the number of clubs he can attend before he gets too full of pizza.

Sort the clubs based on the amount of pizza Blaster will eat at the club, with the lowest amounts first.

While Blaster is under his pizza capacity, look at the club with the next lowest number of pizza slices. If he is not busy at that time, add that number to the total slices eaten.

Print the total number of clubs Blaster can attend.

A=B (Kelly Dance)

Summary

Given a starting string, *s*, and a list of substitution rules. Simulate applying the rules to the string. At each step, find the earliest rule in the list where the left side of the rule appears in *s* and replace the leftmost instance of that string with the right side of the rule. Return to the beginning of the list of rules and continue. If, after a replacement, the length of *s* exceeds 255 characters, immediately exit and print "`Memory Limit Exceeded`". If after 5000 steps the left side of some rule still occurs in *s*, then exit and print "`Time Limit Exceeded`". Otherwise print the final value of *s*.

This problem is really a reading comprehension test. It is difficult to understand what the relevant edge cases here are.

A=B (Kelly Dance)

- In a loop 5000 times, find the first rule that applies and apply it.
- If after any step the string has length greater than 255 characters, output "`Memory Limit Exceeded`".
- Afterwards, if you can find any rule to apply then output "`Time Limit Exceeded`".
- Note that even if that 5001-st rule would cause the string to exceed 255 characters you should not output "`Memory Limit Exceeded`".

Orecart Boba Easy (Kelly Dance)

Summary

Given the ore cart is moving at a constant speed of 1 meter per second along an l-meter long route, n stops that Eugin and Kelly need to make along the way, and v (the maximum speed at which Eugin and Kelly can run), will they be able to reach the end at exactly the same time as the ore cart?

Kelly and Eugin cannot get ahead of the ore cart at any time and must stay at each of the n stops for at least w_i seconds at each boba stop.

They can never *arrive* at a boba stop before the ore cart arrives, but they can *leave* the stop after the ore cart leaves.

Ore Cart Boba Easy

The earliest at which they can arrive at any given stop is given by the maximum of the following two times:

- The earliest time that the ore cart arrives at the next stop
- The earliest time that they leave the previous stop + the time they have to stay at the previous stop + the time it takes to run to the next stop.

Mathematically,

$$
\begin{aligned}
t_0, w_0, w_{n+1} &= 0 \\
d_{n+1} &= l \\
t_i &= \max\left(t_{i-1} + w_{i-1} + \frac{d_i - d_{i-1}}{v}, d_i\right)
\end{aligned}
$$

Ore Cart Boba Easy

Mathematically,

$$t_0, w_0, w_{n+1} = 0$$
$$d_{n+1} = l$$
$$t_i = \max\left(t_{i-1} + w_{i-1} + \frac{d_i - d_{i-1}}{v}, d_i\right)$$

What we really care about is t_{n+1}, the minimum time at which they can arrive at the end of the route.

Thus, if $t_{n+1} \leq l$, then they can reach the end at the same time as the ore cart. Otherwise, they cannot.

Orecart Boba Hard (Kelly Dance)

Summary

Same as the easy version, but instead we must calculate the minium v such that they can get to the end of the route at the same time as the ore cart.

We must find the minium value of v such that $t_{n+1} \leq l$.

We can try all possible values of v and then use the code from the easy version to determine if that is fast enough for them to get to the end of the ore cart, but v is a real number, and even if it were required to be an integer, it would be too slow (TLE).

Instead, we have to *binary search for the answer.*

Orecart Boba Hard (Kelly Dance)

Summary

Same as the easy version, but instead we must calculate the minium v such that they can get to the end of the route at the same time as the ore cart.

We must find the minium value of v such that $t_{n+1} \leq l$.

We can try all possible values of v and then use the code from the easy version to determine if that is fast enough for them to get to the end of the ore cart, but v is a real number, and even if it were required to be an integer, it would be too slow (TLE).

Instead, we have to *binary search for the answer*.

Ore Cart Boba Hard

Instead, we have to *binary search for the answer*.

We can set our high and low values to l and 0, respectively for the binary search.

Try $\frac{l}{2}$ and see if that is fast enough, and if it is, then try $\frac{l}{4}$, or if it's not fast enough then try $\frac{3l}{4}$.

Continue this until the high and low values are close (within 10^{-4}) of one another.

Note: you should first also check whether they can make it in time while going at $3 \cdot 10^8$ m/s and if they can't, then print `IMPOSSIBLE` and return.

IP Matching

Summary

Implement longest prefix matching to match IP addresses to a lookup table.

Two ways to represent IP addresses:

- Integers: use bitwise operations to match
- Boolean array: manually check bits to match

Using strings might TLE because they are immutable.

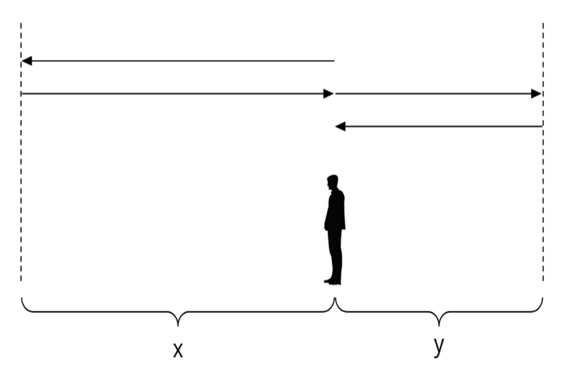
Solution

For each IP address, iterate through the lookup table and find the row that results in the longest prefix match.

Mirror Maze (Nathan George)

Summary

Place mirrors so that your *k*-th reflection appears *d* meters away



Mirror Maze

The first reflection appears $2x$ meters away, the second refection appears $2x + 2y$ meters away, the third reflection appears $2x + 2y + 2x = 4x + 2y$ meters away, and so on.

$$2x, 2x + 2y, 4x + 2y, 4x + 4y, 6x + 4y, 6x + 6y, \cdots$$

Odd k

In general, for odd k, where $k = 2n + 1$, the k-th reflection appears $2(n + 1)x + 2(n)y$ meters away.

Even k

For even k, where $k = 2n$, the $k - th$ reflection appears $2(n)x + 2(n)y$ meters away. We will split this into two cases.

Mirror Maze

We will solve the two cases separately.

Even k

We can simplify $2(n)x + 2(n)y$ to $2n(x+y)$. Note that $k = 2n$, so this can be simplified again to $k(x+y)$. We want to solve

$$d = k(x+y)$$

This is easy

$$\frac{d}{k} = x + y$$

Thus as long as d is divisible by k and $\frac{d}{k} \geq 2$ we can just pick any pair of x and y which adds up to $\frac{d}{k}$.

Mirror Maze

Odd k

We can also simplify $2(n+1)x + 2(n)y$ to $(2n+1)(x+y) + x - y$. Note that $k = 2n+1$, so this can be simplified to $k(x+y) + x - y$. We want to solve

$$d = k(x+y) + x - y$$

This is not easy

$$\frac{d}{k} = (x+y) + \frac{x-y}{k}$$

There are two cases, either $x - y \geq 0$ or $x - y < 0$

Mirror Maze

$x - y \geq 0$

Lets let $x - y \equiv d \mod k$, then we can simplify the expression to

$$\left\lfloor \frac{d}{k} \right\rfloor = x + y \quad (1)$$

As long as $\left\lfloor \frac{d}{k} \right\rfloor > d \mod k$, these are both valid expressions.

$x - y < 0$

Lets let $x - y \equiv (d \mod k) - k$, then we can simplify the expression to

$$\left\lceil \frac{d}{k} \right\rceil = x + y \quad (2)$$

As long as $\left\lceil \frac{d}{k} \right\rceil > |(d \mod k) - k|$, these are both valid expressions.

Mirror Maze

Solution

For even k, the answer exists as long as d is divisible by k and $\frac{d}{k} \geq 2$. Just pick an x and a y such that $x + y = \frac{d}{k}$.

For odd k, check both

- $x + y = \left\lfloor \frac{d}{k} \right\rfloor$ and $x - y \equiv d \mod k$
- $x + y = \left\lceil \frac{d}{k} \right\rceil$ and $x - y \equiv (d \mod k) - k$

Then, solve the linear equations.

Blaster The Daredevil (Kelly Dance)

Summary

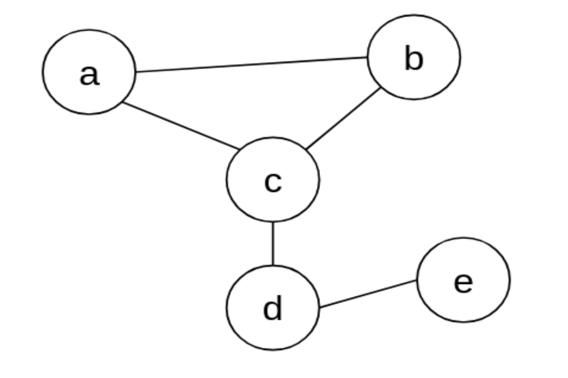
Given a list of vertically aligned floating hoops in the *XY*-plane, choose an angle to launch Blaster such that he passes through the largest number of hoops.

Compute the angle from the origin to the bottom of each hoop, and the angle to the top of each hoop. Put these angles into a list and sort it. Now iterate through through this list, when you encounter the start of a hoop increase a counter by 1. Then you encounter the end of a hoop decrease the counter by 1. The maximum value this counter reaches at anypoint is the answer.

Accomplices (Kelly Dance)

Summary

Count the number of independent sets having size k for $0 \leq k \leq n$.



How many different subsets of nodes are there?

Each node can either be in the set or out of the set, so each node has 2 options. With n nodes there are 2^n possible subsets.

$n = 20$, so $2^n \approx 1,000,000$ which is "relatively" small, so lets just try all of them. For each subset, check whether two people in the set know each other. If not, add 1 to the counter for the set of that size. Just need at most 21 counters.

We can iterate through all of the subsets using binary numbers. A 1 in bit i indicates the i-th person is in the group of accomplices, and 0 means they are not. Reconstruct the set of people and check the condition.

We can loop through these binary numbers from 0 to $2^n - 1$.

Rocky Mountain Road Trip (Kelly Dance)

Summary

Given an $n \times m$ grid of integers, representing heights. In a given move, you can go from your current square to any square that is at most one row away and at most one column away. However, you must move to a square with a different height, and you cannot go to a square at a higher height twice in a row or a square with a lower height twice in a row.

Compute the minimum number of moves needed to get from a given starting square to a given ending square, or report that it is impossible.

Rocky Mountain Road Trip (Kelly Dance)

Summary

Compute the minimum number of moves needed to get from a given starting square to a given ending square, or report that it is impossible.

Define $f(x, y, k)$ to be the minimum number of moves needed to get to the square at row x and column y, where k is either zero if the next move must be an ascent or k is one and the next move must be a descent.

We can therefore run a BFS on the induced graph of vertices from above. This runs in $\mathcal{O}(rc)$ time.

Rivalries (Kelly Dance)

Summary

Given each department's rival, find the minimum number of departments left out after creating rivalry pairs. A Rivalry pair is a pair of distinct departments where at least one department considers the other a rival.

Find an unpaired department which no one considers as their rival and whose rival is not paired yet, and pair them together. Repeat until you cannot find any department to pair. If there is no department to pair, then pick any unpaired department with an unpaired rival and pair. Output the number of the departments that did not get matched.