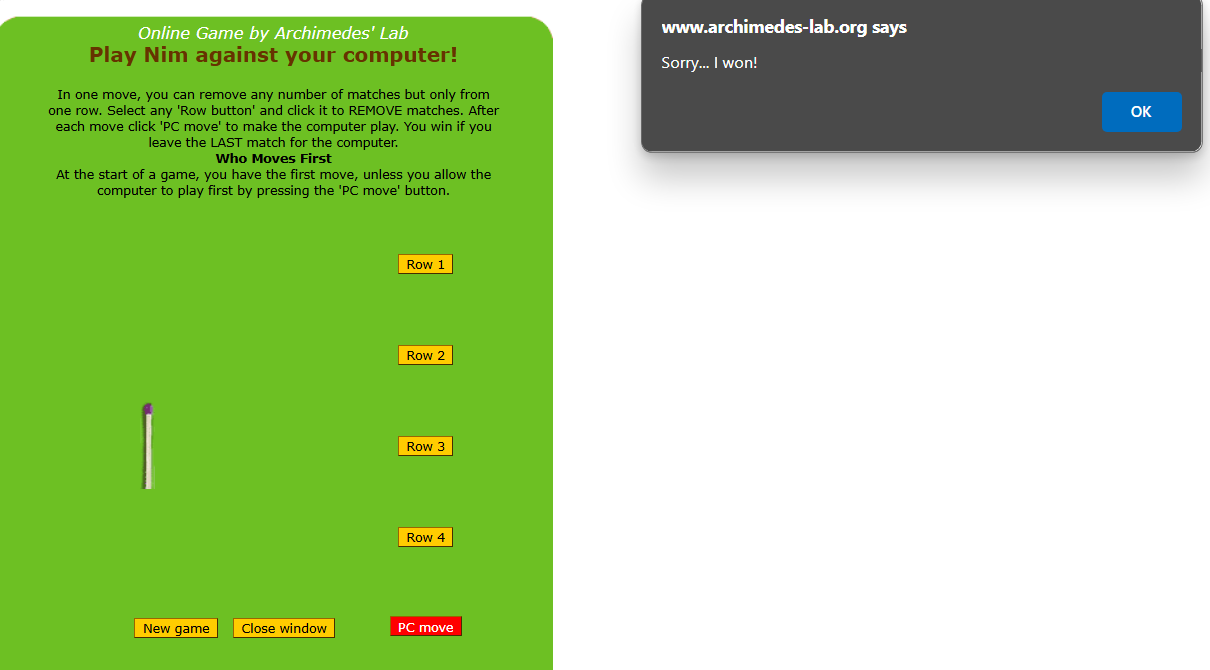
**Game of Nim**

There are certain winning positions that the computer drags you into every time.

* 1 Row: if there are multiple matches and just 1 row, the player moving first wins.
* 2 Rows: 1If there is 1 match in a row and n in another, the player moving first wins. Another fact noted was that the computer tried to match the number of matches (equal pairing) in the remaining two rows before allowing me to move.



Certain winning positions were repeated iteratively, irrespective of my sequence of actions.

E.g. 1 1(computer playing), 2 2(me playing), 1 2(computer playing), etc.

I could not identify any solid pattern for 3 rows, although basic patterns like 1 1 n and 1 m n were identified.

After going through the Wikipedia page, I understood the mathematical concept of ‘Nim sum,’ wherein we must have pairs of 4,2, and 1 match sticks within the rows to attain an ‘equilibrium.’

The 4s, 2s, and 1s must be paired up to attain equilibrium.

This allowed me to answer all the 3 questions:

1. Does a sequence of moves always exist such that the player making the first move wins?

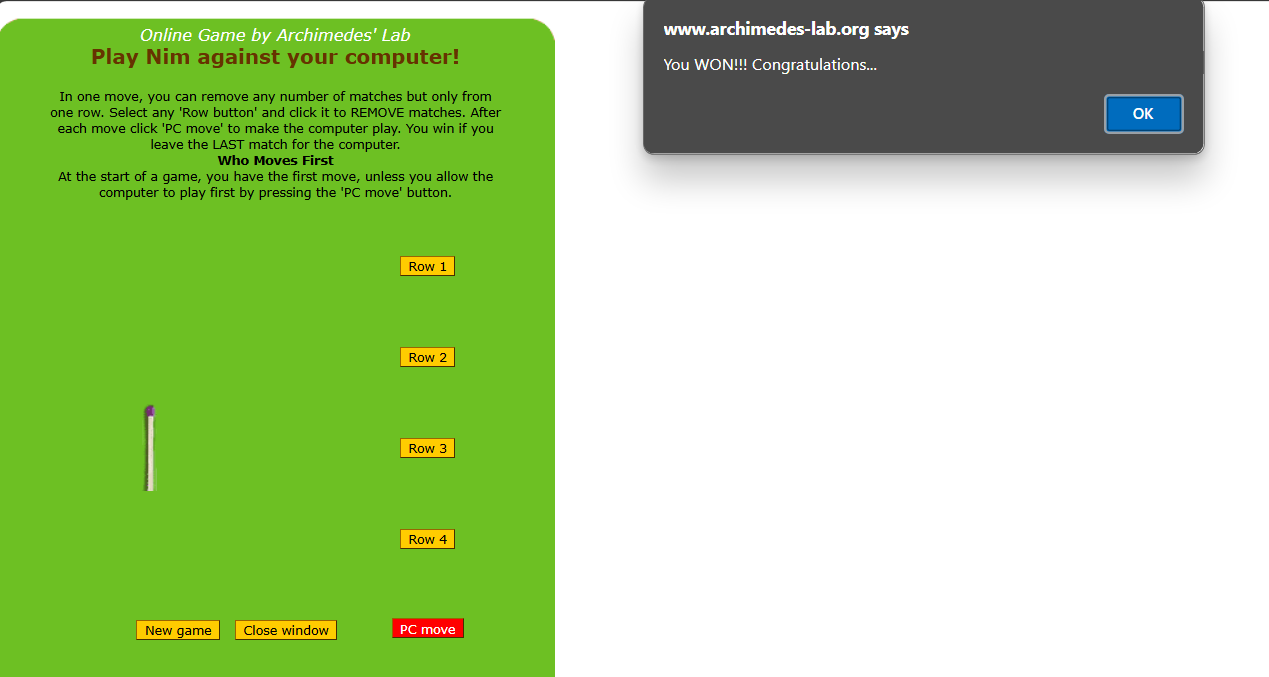
It depends on the setup. In an ideal Nim game environment, it is not recommended to start first because by moving first, you disturb the equilibrium, and the opponent gets a chance to punish you by initiating the equilibrium in his turn, and you end up losing.

1. Given a Nim position, is there a way to determine whether winning for any player with perfect play is possible? If not, why? If yes, how?

The best move can be found for a given position by re-establishing equilibrium. This algorithm can be recursively applied at any given position to win eventually.

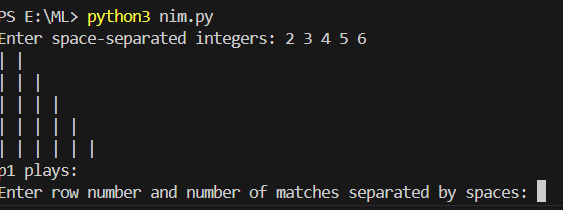
1. If a sequence of moves that guarantees a win exists for a Nim position, how will you determine the sequence of moves? That is to say, how is the computer coded to play the most optimum way?

This can be done easily following the principle of ‘Nim sum.’ We must maintain the Nim sum as zero and only change this for the last move.



The player who is able to restore the equilibrium first (if he does not make any mistakes) is deemed to win the game.

I created a schematic in Python, which accepts the number of elements in each row, and then functions as a two-player game.



The strategy can be coded in to transform this into a one-player game against the computer.