|  |
| --- |
| **Artifical Intelligence Lab** |
| Date: March 21st 2024 |
| **Course Instructor(s)** |
| Ms. Sukhan Amir  Mr. Abdur Rehman |
| **Sessional-I Exam** |
| **Total Time: 1.5 Hour** |
| **Total Marks: 50** |
| **Total Questions**: **02** |
|  |
| **Semester:** SP-2024 |
| **Campus:** Lahore |
| **Dept:** Computer Science |

|  |
| --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Student Name Roll No Section Student Signature |

|  |
| --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Vetted by Vetter Signature |

**Q1: MIN MAX Algorithm [30marks]**

You are tasked with implementing the Minimax algorithm to create an AI player for the game of Connect Four. Connect Four is a two-player game where the players take turns dropping colored discs into a vertically suspended grid. The objective is to connect four of one's own discs horizontally, vertically, or diagonally before the opponent does.

Your task is to write a function or method to determine the best move for the AI player given the current state of the Connect Four board.

Your function/method should take the following inputs:

A 2D grid representing the current state of the Connect Four board. Each cell in the grid can have one of the following values:

0: Empty cell

1: Disc belonging to the first player (Player 1)

2: Disc belonging to the second player (Player 2)

A boolean variable indicating whether it's the AI player's turn (True) or the opponent's turn (False).

Your function/method should return the column index where the AI player should drop its disc to make the best move.

Constraints:

The input grid will always be a valid state of a Connect Four game.

The AI player (Player 1) will always be the maximizing player, and the opponent (Player 2) will be the minimizing player.

There will always be at least one valid move available.

If there are multiple optimal moves, return any one of them.

**Coding Sample**

def find\_best\_move(board, is\_ai\_turn):

# Your implementation here

# Example usage:

board = [

[0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0]

]

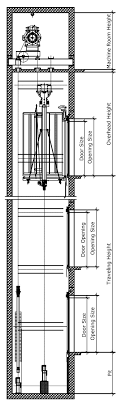
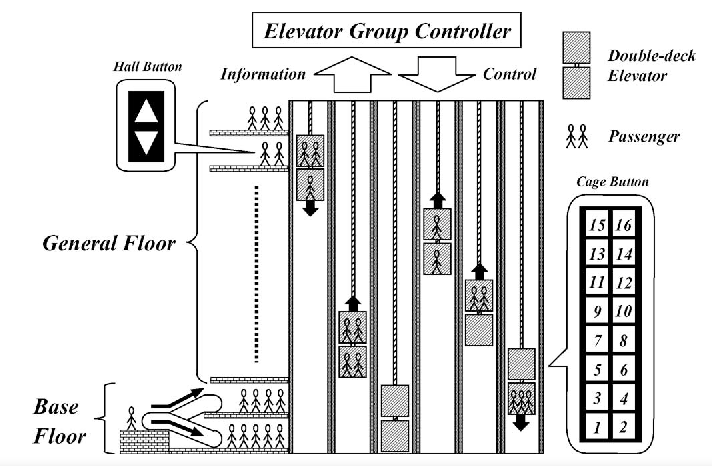
is\_ai\_turn = True

print(find\_best\_move(board, is\_ai\_turn)) # Output: 3 # The AI player should drop its disc into column 3.

**Q2: Anato's Valet Service at FAST NU**

Anato has applied for a job at FAST National University. Since he has zero education, the university has no choice but to offer him a position as an elevator operator for the newly installed elevators at the newly established new building of the newly renovated FAST University, Lahore.

Now the new building has only 4 floors where the elevator is currently operational.

[ note: ignore the images and their details, they are just there to help your brain visualize the problem easily and also cause they look cool ]

You are tasked with designing an algorithm for a simple elevator system for a building with multiple floors. The goal is to help Anato to optimize:

1. **The elevator's movement to minimize the average waiting time for passengers**
   * *since everyone here is in a hurry to complete their assignments*
2. **The total energy consumption of the elevator system**
   * *since FAST is energy efficient university*

The elevator system consists of one elevator that can move in between floors. Passengers can call the elevator from any floor and specify their desired destination floor. The Algorithm must help Anato intelligently decide which floor to visit next based on the current requests from passengers and its current position.

Tasks:

1. Implement a hill climbing algorithm to optimize the elevator's movement strategy.
2. Define a fitness function that evaluates the performance of the elevator system, considering factors such as average waiting time for passengers and total energy consumption.
3. Model the problem space, including the current state of the elevator (e.g., position, direction), pending passenger requests, and the layout of the building (number of floors).
4. Use the hill climbing algorithm to iteratively improve the elevator's movement strategy by exploring neighboring solutions, such as adjusting the order of floor visits or optimizing the direction of travel.