

**Faculty of Engineering & Technology Electrical & Computer Engineering Department**

**Artificial Intelligence-ENCS3340**

**Project #1 : Magnetic cave game**

**Prepared by: Batool Hammouda 1202874**

**Rewa Assi 1201419**

**Instructor: Dr.Yazan Abu Farha**

**Date: 20-6-2023**

* **The idea of the project:**

The idea of the project is to implement a game called “Magnetic Cave” . the game has three modes the first one is playing manually (2 players against each other) the second and the third one is playing against the machine, the difference between the second and third part is who will start the game.

* **How to run the program?**

First when the run button clicked the following tab will appear:

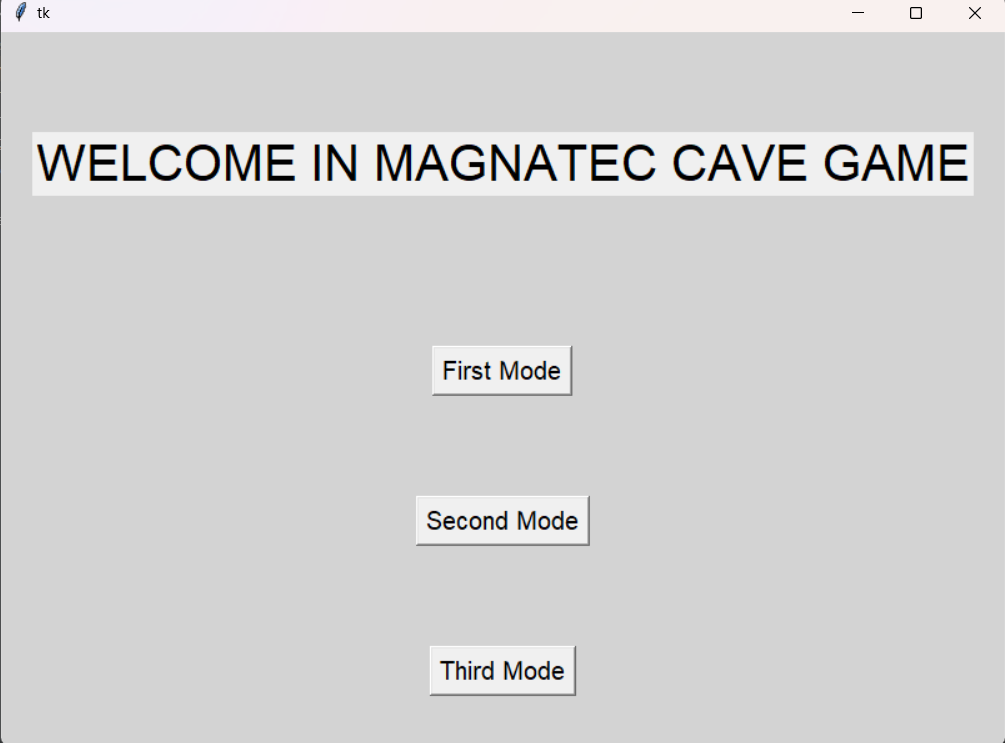


Figure 1:first tab

From this tab the user will choose which mode to run.

* **Case one:** choose the first mode:

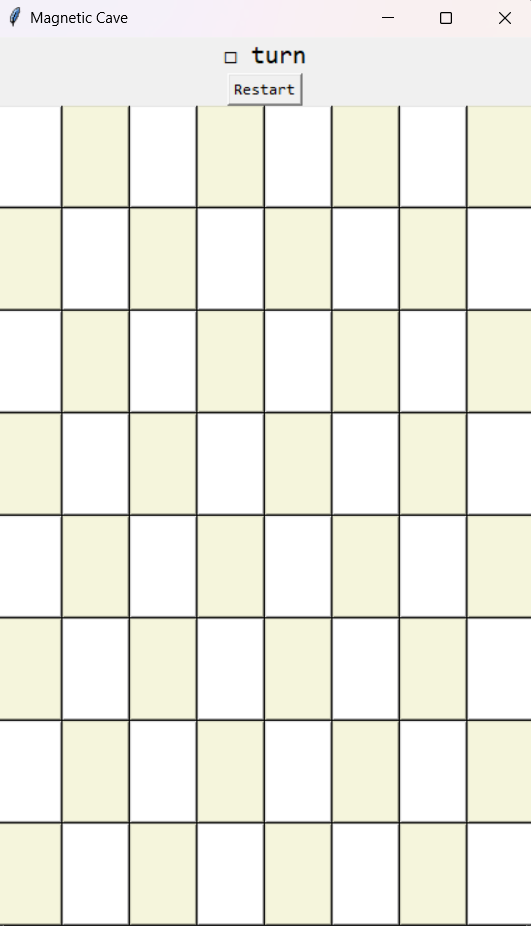


Figure 2:first mode interface

The board will appear to let the two users play, once click in a specific square the small piece will be on the square. the user allowed to play the first move on column 0 or column 7, he is not allowed to play in the center of the bored directly.

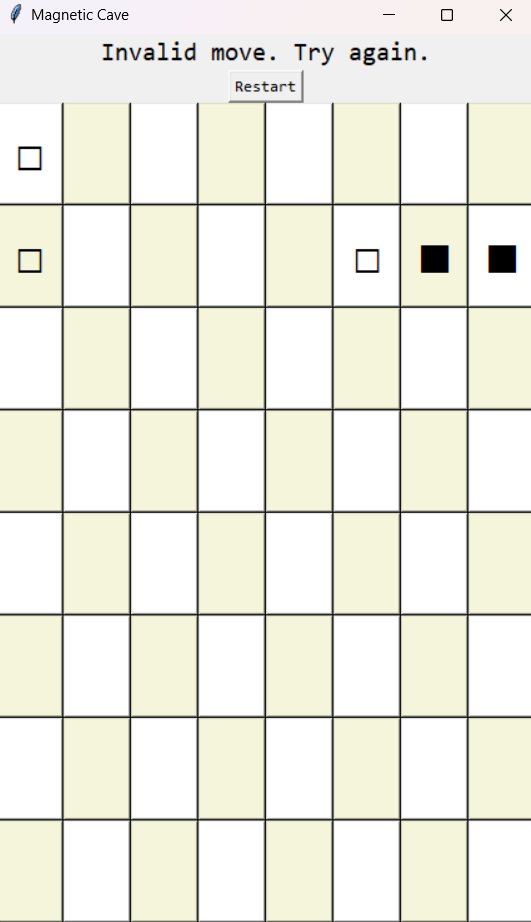


Figure 3:valid and invalid move

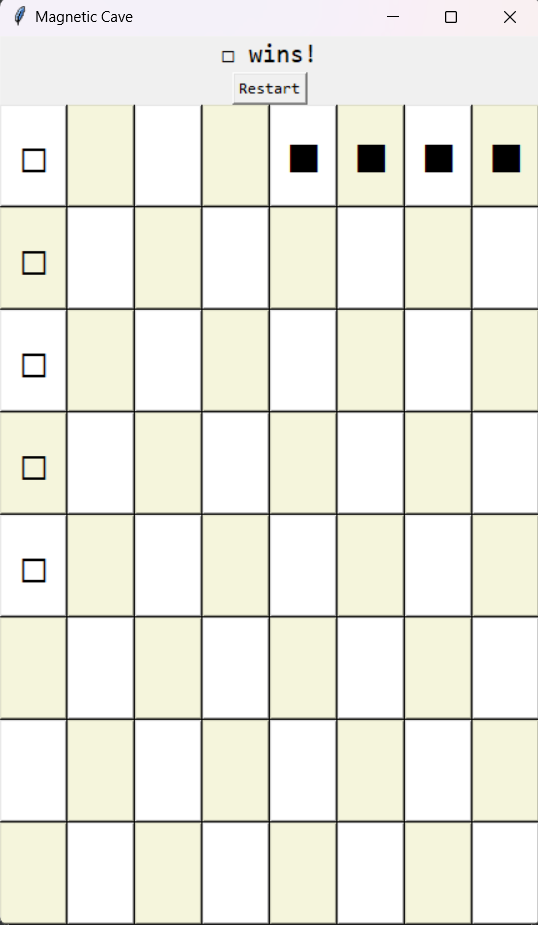
****

Figure 4: win case

* **Case two:** choose the second mode: the black is the user, the other is the laptop

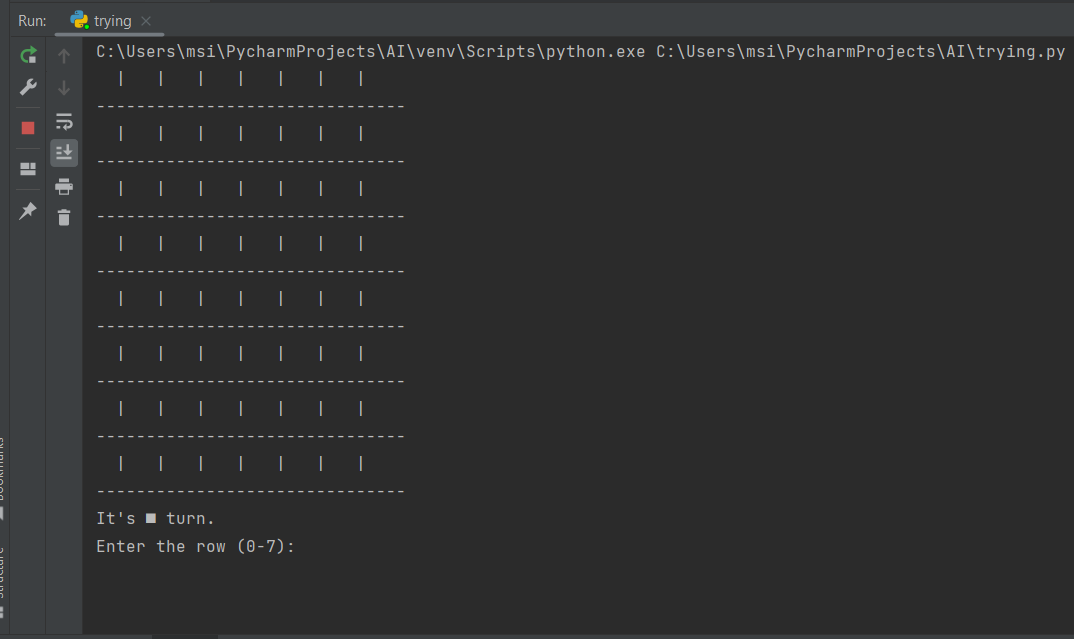


Figure 5:the board initially

* **Case three:** choose the third mode: the black is the laptop, the other is the user

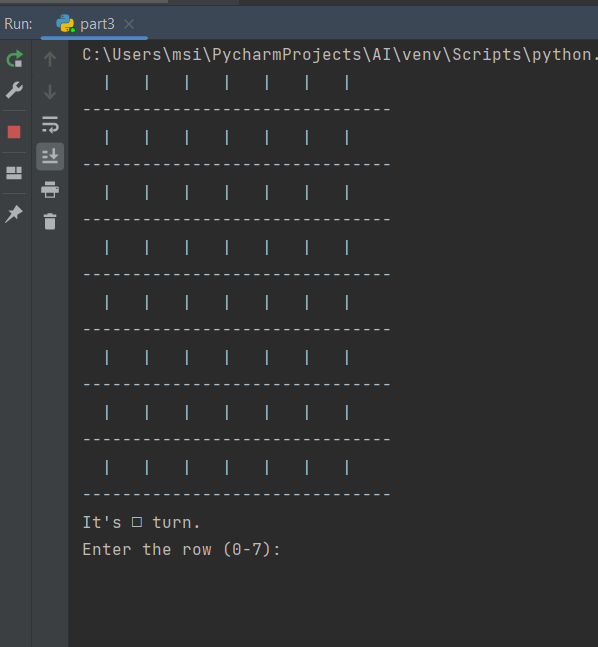
****

Figure 6:third mode

The data structure used in all parts is list to implement 2D matrix as a board. There are some basic functions used in all parts:

* Check winner’s function: function to check who is the winner, it checks if there are 5 squares in beside each other in the same row or column or right diagonal or left diagonal.
* Start function: the first user chooses move then it will check if the chosen move is allowed or not, if not it will give the user a message and the try again to choose another move. the function also checks if the input is empty and handle the exception.
* Evaluate board: is the evaluation function, in order to determine a score, examine the game board's rows, columns, and diagonals. The evaluate window function is used to evaluate each window of the board's components in order to perform this calculation.
* Minimax: The function minimax is an implementation of the minimax algorithm, which aims to find the optimal move in a game by considering all possible moves and their potential outcomes. It alternates between maximizing and minimizing players, exploring the game tree recursively to determine the best achievable score for a given game state.
* Is board full: function to check the tie .
* Compmove: The Compmove function chooses the computer's next move on the game board by evaluating and selecting the optimal move while considering possible future outcomes. It includes a time restriction to ensure that the function is completed on time, which is determined by the depth.

Winning a tournament requires meticulous preparation, adept strategy implementation, exceptional skill, and a favorable dose of luck. Thorough preparation involves acquiring knowledge and training. Effective strategies encompass analyzing opponents and making wise decisions. Skill and competence are vital in executing strategies successfully. Lastly, acknowledging the influence of luck adds an element of unpredictability to tournament outcomes.