Billiards - Style Carrom

Basis:

Carrom is an Indian board game similar in rules to the American billiards or pool. It is played by two players on a square wooden surface smoothened with applied talc, and there are multiple wooden circular pieces involved. At the start of the game, there are 9 black wooden pieces in the middle and 9 white/beige wooden pieces which all surround one red wooden piece in the very center, which is called the queen. A toss or coin flip is typically made before the start of the match which decides who between the two players gets to start. Officially, this may also involve the player choosing the color of his pieces or "carrom men" and choosing which side of the board he/she wants. Gameplay involves the use of a heavier circular "striker piece" which is launched by each player with the flick of his/her fingers. This launch must also occur from a strip of space on the player's side of the board. This occurs in an alternating fashion much like pool, and sinking one of your pieces entails another move for you. The goal of the game is for the player to sink his nine carrom men (pieces of his color) into the pockets with the possible addition of the queen. Each carrom man is worth one point, and the queen is worth three points. Additionally, there is a rule the states that sinking the queen is not sufficient to obtain the points, but rather the player must also follow that move by sinking one of his carrom men. Much like pool, deductions of one point will also occur for sinking the other player's carrom men (the pieces of the other color) and for sinking the striker piece. The goal of the game is to rack up more points than your opponent typically by successfully sinking the queen. The game ends when all pieces have been sunk.

Here's some more information:
https://en.wikipedia.org/wiki/Carrom
https://www.youtube.com/watch?v=5FEv0xLqwo8

The reason my project is named “Billiards - Style Carrom” is that I will be changing some rules to allow for a unique version that is definitely not already out

there. Firstly, I will change the rule whereby players must launch from their own "strip" and make it more like pool wherein each player must shoot from the position where the opponent's striker on the last shot ended up. This is to allow for greater algorithmic complexity and unique gameplay. Additionally, I may make some changes to the point system, and I will ultimately have to implement my coin toss / color pick system in a way unique to the game.

Project:
I intend to create a game as detailed above. In my competitive analysis, I will further detail how my game will be unique with respect to the existing competition, but already you can see that it will be a "Billiards - Style Carrom."

I intend to create the game as follows. The game will have a physics engine to run the gameplay, an AI to allow the user to play against the computer with varying levels of difficulty, and the possible addition of sockets to allow for multiplayer gameplay. The game will have to be broken down into multiple layers. I will list them below.

1) Tkinter graphics with various game states (start screen, gameplay, endscreen), the option to choose an AI of varying difficulty, and gameplay allowing two players on one machine.
2) The definition of the back end and of the frame of the game with various classes such as Piece, CarromMan, MovingCarromMan, and Striker. The game mechanics will also have to be defined here. This will involve removing sunk pieces, counting points, deducting points, returning the queen to the center upon a failed follow up shot, etc., etc.
3) The creation of a physics engine to model pieces hitting one another, and the striker hitting multiple pieces. This will present the first algorithmic challenge. Here, we must define outcomes for the angle of the collision, the speed at which both pieces were moving, what kinds of pieces were colliding, how many pieces collided, and if the collision occurred against the wall of the board, etc. etc.

4) The creation of an AI to play against the user. This would pose the second and greatest algorithmic challenge of the project. This AI would have multiple difficulty levels against which the user could play the game. The AI would additionally have to think not only about how to shoot the striker such that it sinks the maximum number of pieces, but also how to position the striker at the end of its movement such that the opponent will have trouble later in his/her move. It would likely go off of the model of other game AIs, at least at a superficial level; it would gauge the possibilities present on the board and make a tree to decide the most advantageous solution.
5) The possible implementation of multiplayer with sockets.

Plan:
The physics engine and game mechanics will definitely be the first hurdle. Although, as of now, I imagine that they can be accomplished. It will come down to modeling the collisions correctly and reading about aspects of momentum and inelastic and elastic collisions. I would go about this specifically with timer fired to check to see if any of my pieces are moving and if their areas are crossing over. From here, I would check to see how they are oriented, and how they are moving to determine what will happen. The basic principle at play here is conservation of momentum. One troubling aspect, however, is how to model collisions between multiple pieces and how they will react. Other miscellaneous elements will be modeling friction as pieces will eventually slow down and stop.

The actual challenge will be the implementation of the AI. If the above paragraph turns out to be too much of a hurdle, this may have a smaller implementation if one at all. However, I imagine that this too will be feasible. Understanding how Game AI works in this scenario would be difficult, and I imagine that I may need TA assistance on some parts of it. However, it would follow a similar model. It would construct a tree of possibilities of moves to sink pieces into their positions. On harder difficulty levels, the computer would simply know how fast and at what angle to launch the striker piece to get a carrom man into a pocket. However, it must consider ways of landing multiple pieces, with a special emphasis on sinking the queen with a follow up piece. I would have to make it consider how to possibly

ricochet multiple pieces At lower difficulty levels, I could introduce varying levels of randomness to ensure that the computer behaves oddly and makes mistakes.

Modules:
I will only be using Tkinter for my game to maintain algorithmic complexity, and for stylistic reasons.

Other python libraries like string, random, etc. might be used.

Update 1:
Edit: The implementation of the AI will follow a different approach that will allow for a more playable experience. The computer can perform well, without necessarily needing a full heuristic model. This will also allow for a more streamlined approach to creating levels against which the player can play against the computer.

> **After meeting:**
> Monte - Carlo / Minimax
> Make the striker AI piece make predictions.
> In this last week, implement the hard level fully.
> Improve game aesthetic

Update 2:
The AI has been developed to a level that is within Tkinter's capacity and allows for an enjoyable gaming experience. The shot engine, while weakened by Tkinter's imprecision, performs well and makes accurate shots. The algorithm currently emphasizes pieces that are closest to the pockets, and the shot engine is able to use the necessary physics to sink the piece. Other features have been added to the code as well, but are not fully implemented within the game itself as it pushes the bounds of what Tkinter can run. These include an algorithmically more complex version of the AI, and an option to use your mouse to create a line which shows the user the direction in which he/she is shooting.