Updated Design Doc below:

TP3 Updates:
My physics engine with collision and friction is working properly. My 2 player aspect of the game was also implemented after TP2 (5 shots per player turn). I updated the visualizations for my title screen, directions screen, and game winner screens. I also added a challenge mode that is a single player version of the game in which the user is racing against a timer. If the user can get every ball (other than the cue ball) into a socket before time runs out, the player wins.

TP2 Updates:
My physics engine is now functioning properly with my collisions and friction forces acting on the balls. The velocity of the cue ball also depends on how far the cue stick is being dragged back (the cue stick is still in progress because it glitches after the first play). There is a home/title screen that allows for the user to either read the directions of the game, or start the two player version. The balls also disappear if they "fall" into the hole on the pool table.

What do I still need to include?
I still need to include winner conditions, and alternate turns between two players.

New Timeline:
12/3: fully functioning two player game without any bugs
12/4 - 6: research minimax and attempt to implement into my program

Updated Structural Plan:

→ appStarted(app): initializing variables
→ isHit(app, ball, ball1): checks to see if a collision occurred
→ collision(app, balla, ballb): updates new velocities and angles
→ socket(app, ball): checks to see if any balls fall into sockets
→ move(app, ball): updates positions of the ball
→ powerCueHit(app): updates stick positions based on mouseMoved and mousePressed x & y
→ mouseMoved: circular motion of mouse as it moves around the ball
→ mousePressed: gets x, y of mouse when clicked
→ mouseReleased: gets, x, y of mouse when released
→ keyPressed: takes user to direction page (still need to create this page), and main screen
→ timerFired
→ redrawAll
    → drawTitleScreen
    → drawMainScreen(app, canvas)
        → drawTable
        → drawBalls
        → drawStick

TP1 Updates:

I don't really have any significant updates to my design documents except for a shift in my timeline. Due to my busy athletic schedule last week, I am a little more behind on my progress than I would like to be. I worked on more of the graphical aspect than expected, but still need work on completing the actual physics behind my engine. My new timeline is below:

11/23 : completion of accurate Physics Engine
11/24 : cue stick function to move the balls
11/25 : two player functionality of the game
Basic MVP completed before Thanksgiving break is over.
11/26 - 30: research minimax and attempt to implement into my program

My structural plan is also updated below:

Ball Class: creates ball objects
Vector Class: calculates vector math
Stick Class: creates cue stick
Physics Engine:

- isHit(): checks to see if a collision occurred
- collision(): should calculate new angles, velocities, acceleration, and positions of the ball
- bounceWall(): changes direction of ball if it hits the edge of the table
- socket(): removes ball from the game if it "falls" into one of the holes
- move(): will properly update position, velocity, acceleration based on forces acting on ball (like friction)

Graphics:
Within redrawAll function –
drawTable()
drawBalls()
drawStick()

8-ball:
This game is going to be an accurate version of pool that incorporates collision, friction, and spin, and power of the hit. There will be a two player game option where the players take turns trying to get all their color balls in one of the six holes on the pool table with the offered virtual pool stick. The player to get all their color balls, AND the black ball first will be the winner of the game. There might also be an AI tool that is able to find the best moves, and allows the user to play against the computer.

Similar Projects:
There are many different versions of the game, 8-ball, online. Many of them incorporate elastic collisions to calculate the position of the balls after a hit, and many versions offer 1 and 2 player

versions of the game. The single player version uses AI to calculate the best possible moves in order to play against the actual user. I plan to use both of these features in my program.

Many programs that I have seen online calculate hits that are based on if you hit the ball straight in the middle. The goal for my program is to also add a spin feature that allows the player to choose where they want to hit the ball (ie the top vs the bottom). I am hoping to add more “bonus” features to my program, but I am not yet sure what I want to incorporate yet (maybe a feature where you can earn coins/points to earn hints about best moves to make).

Structural Plan:

Vector Class: This class helps with vector calculation for my physics engine.

Physics Engine Class: My physics engine will create my ball objects, and calculate how the collisions, friction against the table, and spin affect their movements. I will establish position, velocity, acceleration, and any other forces necessary in this class.

Key Pressed Function: This function allows the user to drag their mouse back from behind the ball and release to hit the ball at the assigned angle. The amount of force applied to the cue ball will be determined by how far the user drags their mouse backwards. This function should also allow the user to choose spin based on the tool offered on the screen by clicking which part of the ball they want to hit.

AI move class: A class to determine which next move is the most beneficial for the player to take. This will allow me to create a 1 player version of the game in which the user can play against a computer.

Redraw function: This function will visually display a 2d version of the pool table, along with all the ball objects that will be created in my physics engine class.

Algorithmic Plan:

I think the trickiest part of the project will be to create my physics engine class. If I am able to calculate how each ball object is supposed to move, visually displaying those objects should be easier to do.

Within my physics engine class, I plan on creating vector objects to keep track of the balls’ positions, velocity, and acceleration. Using these vector objects, I will be able to calculate the starting velocity of the ball based on the power of the initial hit, the elastic collisions that affect the movement of the ball (m1+vi + m2v1 = m1vf + m2vf), the force of friction between the ball and the table and how that slows the ball down to a stop, and the spin and how hitting the top vs the bottom of the ball affects the movement of each object. Finishing my physics engine will allow me to focus on the graphics, and implement how the game should really be played.

Friction: The sliding friction of a real pool table has a relatively small coefficient of 0.06, so I am going to multiply this coefficient by the Normal Force to calculate the friction slowing the balls down.

Spin: I think spin will be the most difficult to calculate because I'm not really sure how the physics behind that really works. I am going to need to research and look more in depth about this aspect.

I also think it will be difficult to keep track of and account for each and every collision happening at the same time. I am going to need to keep track of the position of every ball object, and determine how much force is being exerted through every hit of contact (I will probably need to create a list of positions, and check through this list of positions as the ball moves?).

If I am interested in making my game more complex, I will also need to look more into game AI and how my program can predict the best next moves to make. This will probably increase the complexity of my program, but I still need to do more research on how game AI actually works.

Timeline plan:

11/21:
- Completion of physics engine

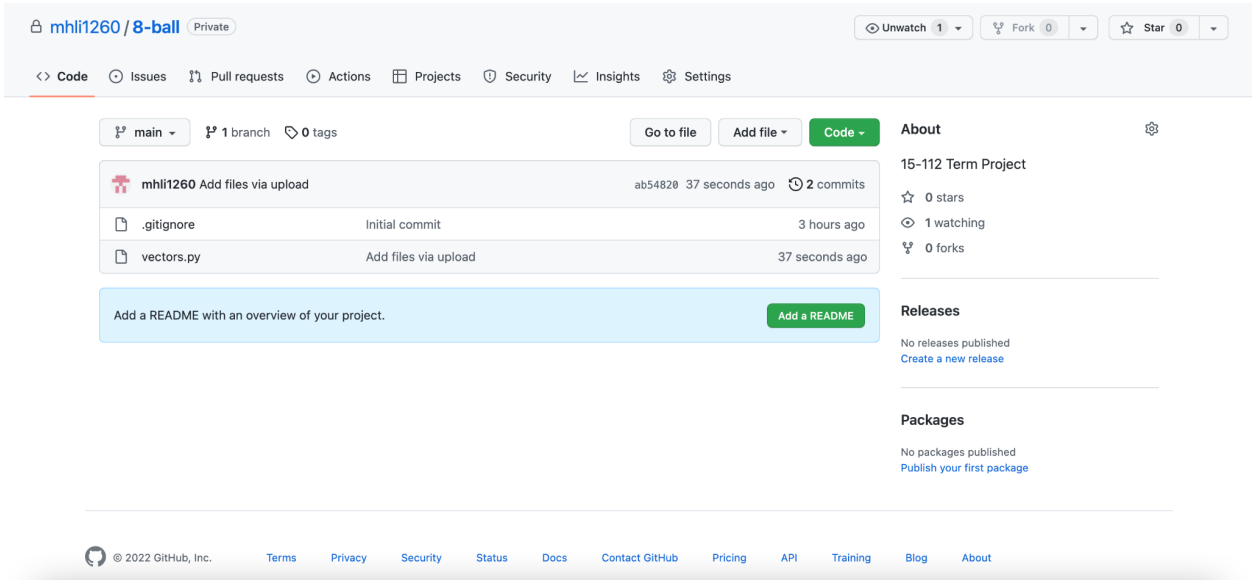
11/24:
- Graphics
- Functioning 2 player aspect of the game

11/27:
- AI move prediction algorithm
- Functioning 1 player game

Basic rough draft should be done by the end of thanksgiving break!

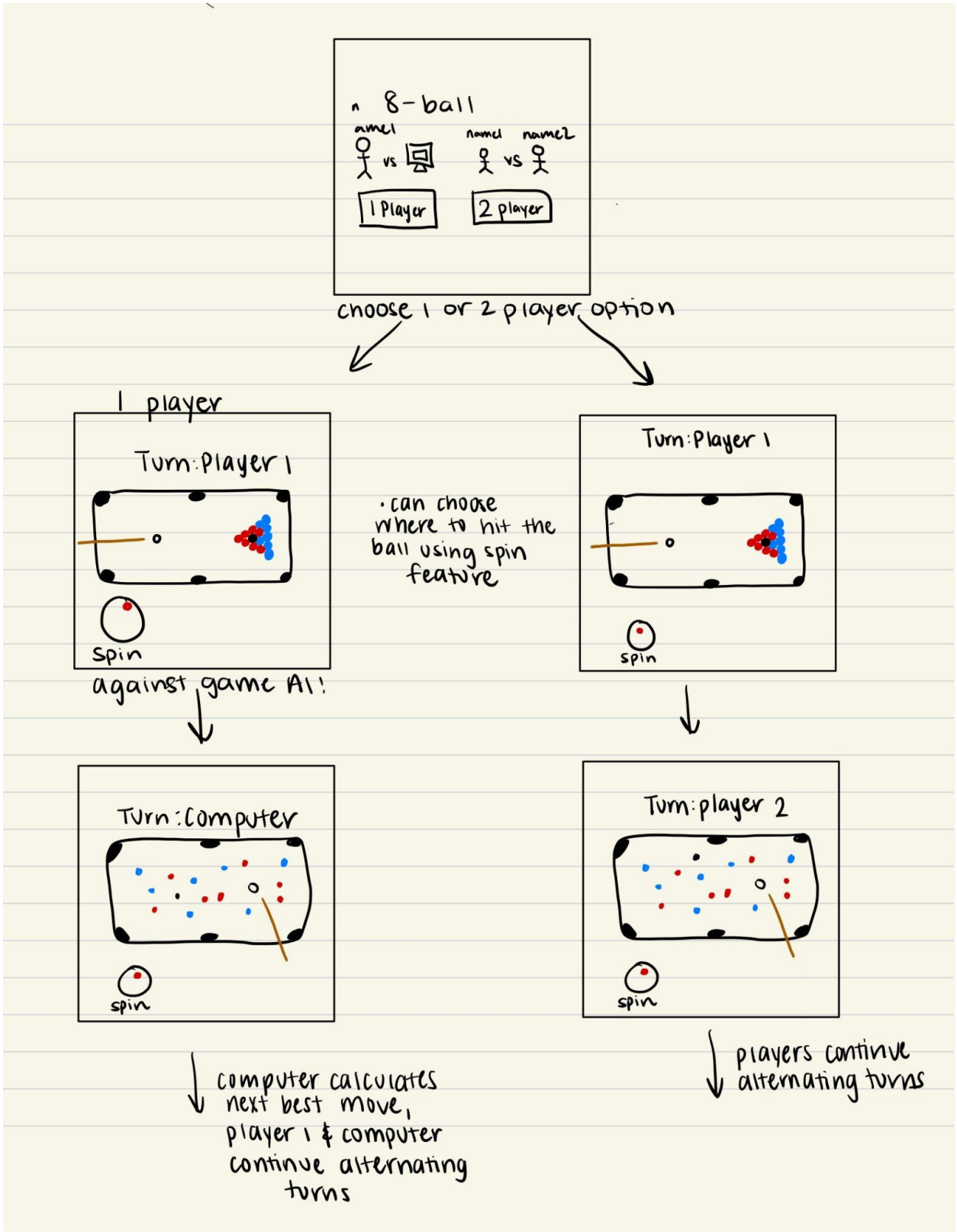
11/27-12/8:
- Time to fix up bugs!
- Extra features and fix-ups should be worked on until 12/8
- Make sure the program runs smoothly and efficiently

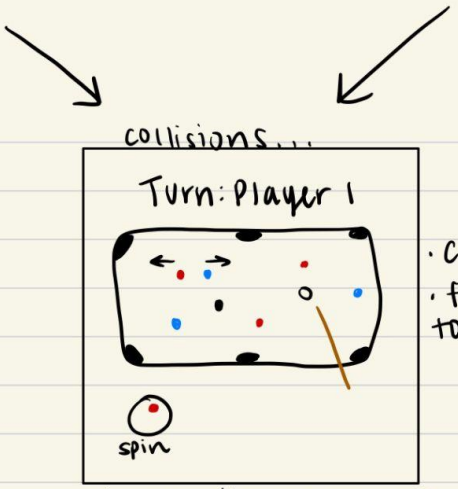
Version Control Plan:
I plan on using github to back up my code. So far, I have created a new repository for the project where I will upload my code as I write it. I currently have my vector class uploaded. This will ensure that I have it stored somewhere in case my program crashes.



Module List:
I don’t plan on using any additional modules.

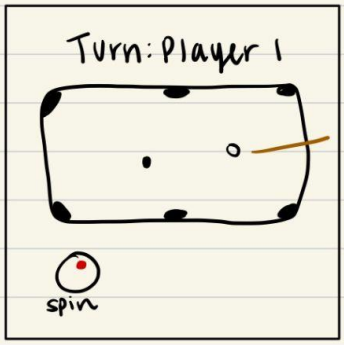
Storyboard:


8-ball
name1
vs
name1
name2
vs
1 Player
2 player
choose 1 or 2 player option
1 player
Turn: Player 1
spin
against game AI!
• can choose where to hit the ball using spin feature
Turn: Player 1
spin
Turn: Computer
spin
computer calculates next best move, player 1 & computer continue alternating turns
Turn: player 2
spin
players continue alternating turns

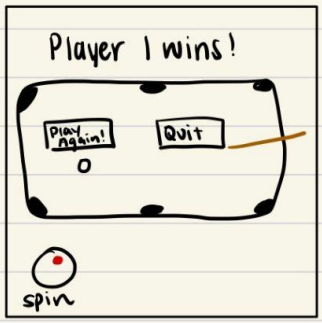


- collisions when balls hit
- friction on table brings balls to a stop

balls disappear after "falling" into 1 of the 6 sockets



last player to get rid of all their color balls & the black ball in a socket wins the game!



2 options after establishing a winner: play again or quit