Game Reports

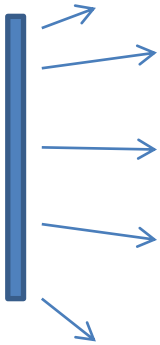
By: Owen Austin Oei

ID : 30039096

1. Summarize the workings of the code and highlight the interesting parts.

    This game's implementation was heavily relying on how it was implemented in the asteroid game by Tim in https://tgdwyer.github.io/asteroids/. The game was saved in an object named state and edited in every step using tick. These things will be further explained later in the report.

    The most interesting part of the code is how the ball bounce of the paddle and the if statement made with the question mark and the double colon. I made the bounce on the paddle with a ratio, the more edge it is the more tilted the path of the ball will be, as shown in the diagram below



    Another part is creating if statement with a question mark is quite new to me so I find it really interesting. This is a snippet where I find it very funny.

```
const ultStat = (s:State) : State => s.paddle.ult_stat == false? s.time - s.paddle.createTime  >= 2000? {...s, paddle :{...s.paddle, ult_stat: true}} : {...s}
:s.comp.ult_stat == false? s.time - s.comp.createTime >= 2000? {...s, comp:{...s.comp, ult_stat:true}}: {...s} : {...s}
```

2. Explain how you tried to follow the FRP style.

    I tried to refrain from imperative way of writing the code. Using observable streams instead of using document.onkeydown and stuff. Other than that, we can see it is frp when there are no mutable variable being used in the whole program. We also can see

that the data we are storing are only the current data as a whole and changing it by using a function instead of using classes and object like in Object Oriented Programming

3. How you manage state throughout your game.

So I save an object with type state that contains every information about the game. Every ten milliseconds, or 0.01 seconds, the system checks if there is any button pressed. If there is any, do what the button tells you to do. If not, we need to iterate the game meaning moving the ball into the next direction based on their velocity and acceleration stored in the body type. The body types stores all information regarding a certain object in the game.

Every action done have a certain object of class created for that action. If there is no action the class tick will be created, in this class all the stuff need to be done(as said above, ex: next position of the ball, if hit the wall, etc) will be called.

All the back end of the program, will be done in the reduce state. And all the front end, regarding the graphics and images, will be done by the update view function.

Extra Functionality

I never feel satisfied with a normal pong game against a computer. So I implemented a two player pong game that can play against each other. One use W and S to go up and down, the other one use Arrow Up and Arrow Down key to move up and down.

Other than that, I implemented a unique skill that wills teleport the ball instantly into the enemy area near the goal. I also like the idea of being it a random placement of height, so it can appear on the top of the board or even at the bottom. I also fond of getting the idea if you are not lucky, even your ultimate will not go your way as if the ball appears in the front of your enemy. The ultimate itself will be ready every 20 seconds. The computer cant use the ultimate. The ultimate count down will be counted after the ultimate use.

Blue Circle : Ultimate Status.