Project Proposal

Project Description: I have yet to decide on a name but the project will be a speed running game. Basically, you will compete with AI's or other players to run around a course, the game window is determined by the person in first and if you fall out of the view of the game window, you lose. The game will include power-ups, boost, grappling, wall jumping and other cool skills.

Competitive Analysis: There are many versions of this type of game online. One large way that it is different is that it is not a side-scroller per se. Rather, there is a multi-layered map where you go in circles. While the visual aesthetic is still similar to a side-scroller, players will be rewarded for knowing the map and there can be multiple paths in the map. Another difference is that in this game, the players will be able to interact with other players on the map. They will have the ability to tackle, fire rockets at, grapple the player rather than the wall in addition to more attacks.

The game that this most resemble is a game called speedrunner. The basic gameplay is very similar with a set map, somewhat destructible environment, and grappling and power-ups. Though, the main way in which I will be differentiating this game is by adding interactions between the players and more interactions with the environment.

Structural Plan:

1. Player Class:
    a. Controls the drawing of the player
    b. Moving of the player
    c. Handles grappling
2. Map class
    a. Stores data about nondestructible elements of the map
    b. Handles drawing map
3. Powerups class
    a. Handles location and collision between player and power-ups
    b. Manages when power-ups are used and when they run out
    c. Manages effects and visuals of power-ups
    d. Draws power-ups
    e. Part of the destructive environment as when you pick up a power-up, it is removed
4. There will be a periodical function that calls the update method on all of the classes to update position and characteristics, the player moves first and then

the rest of the things react to that movement. Then the periodical calls all of the draw methods, starting with the map, than powerups, and then the player
5. There is also a function that moves the screen to keep it in the correct position

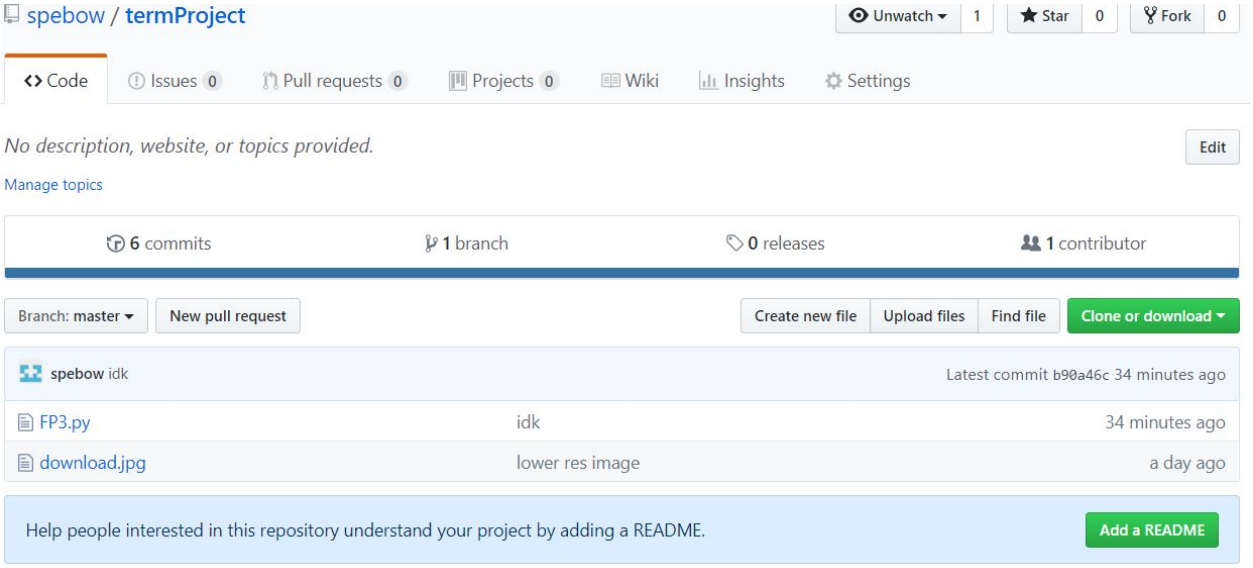
Algorithmic Plan:

1. Grappling
    a. I already have the math so that if I give it a delta angle, the player moves accordingly, rotating around a fixed point, though that delta angle is constant right now and not tied to physics. I plan to use physics equations to calculate the x and y components of acceleration at that angle and use that to change x and y speeds which is what changes the position.
2. Wall collisions:
    a. Right now single wall collisions work correctly though double wall collisions are a little more complicated. When it hits two walls and the walls are concave, it places them against both of them. By concave, I mean the walls on the corner of the map where the player can touch both walls simultaneously. For convex walls, I calculate which wall was hit first and have it collide with that wall but not the other wall as it would in real life. Well I think this is the right track, it is still a little buggy
    b. In addition, I would like to add support for angled walls/floors,
3. AI players:
    a. Right now I have no code to allow or deal with a second player so I would have to write that in addition to having to make the AI actually move
    b. My plan for making the AI move is to have it be attracted to checkpoints along the map and repulsed from walls, in addition, he would be attracted to powerups. This is the part of my project I am least confident in
4. Physics not tied to FPS?
    a. This one I don't really have a plan to implement but I am kind of scared of my game slowing down randomly during intense scenes. Right now, if my fps is cut in half, so is the game time. Like the speed of the character, the gravity, everything is based on fps. That is fine if the fps stays constant but gives the game a weird feel if the fps drops

Timeline Plan: By TP2, I should have finished my MVP. For that, the largest thing that I have to do is add sprites and visuals to my game. In addition, I would like to have a fully fleshed out power-ups suite in addition to support of a second player. While I would love for that player to be ai, it would be fine if it was a second human controlled player. The first thing I am going to work on is sprites as that is what I am least comfortable with and most necessary to have a MVP. I am also scared about how that will affect the FPS of

my game and want to implement that in first before extra features that will further slow down my game. After that, I will add AI and second player support. The last thing I will work on is a suit of power-ups. The game already has boost which is the most important one. While I would love a lot of power ups, they are the least necessary part of the game.

Version Control Plan: I have a GitHub repo (https://github.com/spebow/termProject) that I can use for version control and backing up my work. I have used GitHub before.



```
C:\termProject>git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   FP3.py

no changes added to commit (use "git add" and/or "git commit -a")

C:\termProject>git add FP3.py

C:\termProject>git commit -m "changed grapple extending speed/logic"
[master f6365bb] changed grapple extending speed/logic
 1 file changed, 1 insertion(+), 1 deletion(-)

C:\termProject>git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 326 bytes | 326.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/spebow/termProject
   b90a46c..f6365bb  master -> master
```

Module List:

1. Pygame
2. Keyboard
3. Sys
4. Time
5. Random
6. math