Project Description:

TP Name: Starry Planet

Description:

You are a newly-born asteroid on a gigantic planet. Fortunately, the starry sky projects its constellation as the Starry energy onto the ground. You must catch the Starry energy and extract it to evolve and gain mass. There're dangers behind you, and any Starry energy you miss will transform into your enemy because the starry sky is pissed off. Remember the rule: Eat the food, Don't get caught by the food, and Gain your mass!

Competitive Analysis:

The original inspiration for my term project is a similar parkour game Kreator (https://www.mobygames.com/game/kreator). But as a commercial game, Kreator puts a higher emphasis on the delicacy and beauty of the game, and in turn, simplifies the coding difficulties of it. For example, the "move" players can move is only up and down in vertical direction, the food positions are hard-coded into the map, and the sky and the earth are image-based sprites.

So I want to make the game more sophisticated in the aspect of programming rather than aesthetic beauty. In my TP, the way players control the ball is not limited to Up and Down. Instead, they can use spaces to move it vertically as original, and they can also use mouse press to pull the ball towards that point. Furthermore, I choose to use the Boids algorithm and Perlin noise to randomly generate food positions and terrain with infinite possibilities. Since I can't find a good sprite for the earth, I decide to use Perlin noise to generate a simple but good-looking sky with blinking stars and constellations, and a earth.

Structural Plan:

The finalized project will be a TP folder containing subfolders: *code* and *resources*.

In the *resources* folder, there are several subfolders: fire and *initialFire* containing the sprites I use for the protagonist. *Tree, flower*, etc are folders in plan, which contains the sprites for trees and flowers.

In the *code* folder, the files are: asteroid.py, cmu_112_graphics.py, perlinNoise.py,

simpleNoise.py, physics.py, and TPMain.py. In the asteroid.py file, I have a class Asteroid which is the class for the player, the food, and the earth. It also has a boids class (inheritance of Asteroid) which specifies the behaviors of the food. In the perlinNoise.py file, I have an implementation of generate Perlin noise. In the simpleNoise.py file, I have an implementation of generate simple noise. In the physics.py file, I have several physical rules that construct a simple 2D physics engine for the world. In the TPMain.py file, this is the main entry for the game, and it modulates the parameters of the game.

Algorithmic Plan:

I think the randomly generated food positions and the terrain is the most complex part. I need to use Perlin noise for a continuous altitude of all the flocks of food, and I need to give the food some velocity and acceleration to enable its elliptical motion. I also need to implement the Boids algorithm on it, to ensure that the separation of the food is enough.

The 2D physics engine is very intricate, and all the physics applied to it need to interconnect with each other. I think the 2D collision part is the most complex part of 2D physics.

The earth are generated using Perlin noise, and the way I draw sky line is reject sampling.

The sky and the stars are drawn using simple Noise. It allows more sudden variations than perlin noise, which is better for the beautiful coloring of stars.

Timeline Plan:

TP1:

2D physics engine

Perlin noise for sky and color

A crude way to randomly generate food

How to eat the food or turn into enemy and a scoreboard

TP2:

A mature way to randomly generate food

Terrain generation which sets several stages of the game. Eg: grassland, forest, dessert, ···

(Mostly graphics)

A welcome page, and a setting page which give the player more instructions on how to play it, and allow the user to change some of the parameter in game.

Some special food/enemy with special effect.

An animation of how the enemy will chase after the player.

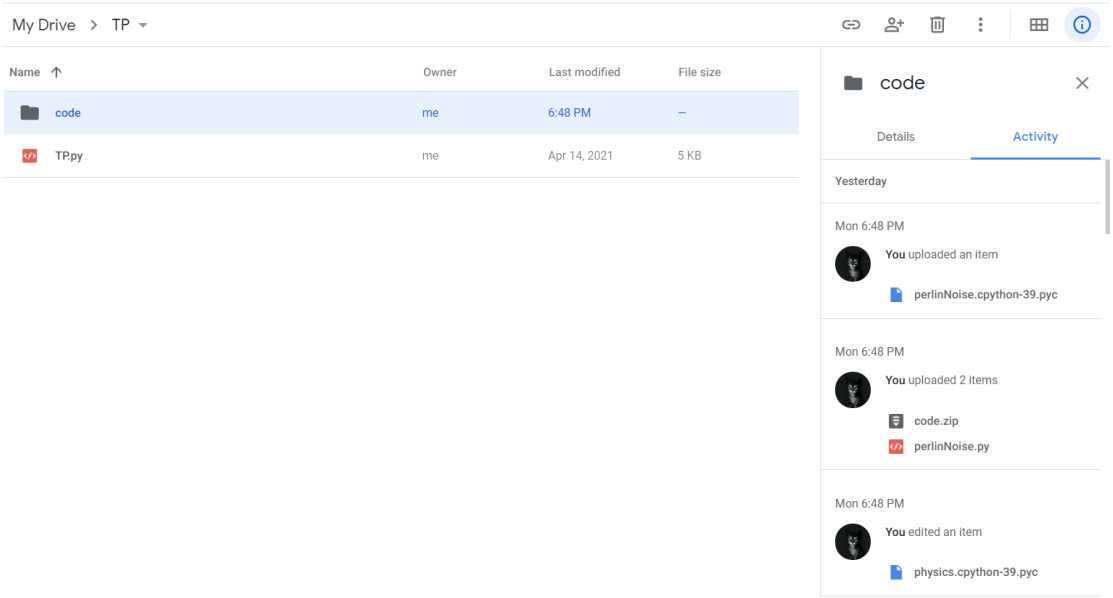
TP3:

Maybe make the game into 3D

Give it a bird-eye view

Version Control Plan:

I choose to back up my code by uploading the code folder to my google drive.



The TP.py file is uploaded before TP0.

Module List:

Numpy

PIL

TP2 Update:

Progress since TP1:

A good UI that contains intro page, story (tutorial) mode, and game mode

A button that is very interactive

An story page that tells the user the rule

Some animations

Implement the Perlin noise by myself

Implement a mechanism for check game over

TP3 Update:

Make the animation of the enemy chasing the player

Food turns into enemy and merge into the large enemy

The enemy chases the player

Interesting conversation between the player and enemy

Re-make the distribution of stars with simple noise instead of Perlin

Add music and soundbar in the setting

Add the record function to the game