

Project Description

Ticket to Ride

This term project is Ticket to Ride inspired from the board game Ticket to Ride. It is a Python simulation game using user-input versus game AI. The premise of the game is for players to collect and play matching train cards to claim railway routes connecting cities throughout North America. The longer the routes, the more points.

Competitive Analysis

From the term project gallery, there aren't previous versions of Ticket to Ride. However, similar projects would include those who made Settlers of Catan. My graphics are similar to the style of Angela Zhang's Settlers of Catan graphics. The initialization and the help screen are similar. Conceptually, Ticket to Ride and Settlers of Catan are different games so that is different between the two.

On Youtube, there is a video "Ticket to Ride - #1 - The Train Board Game! (4 Player Gameplay)" that has many different qualities to my project. The game shown in the video has many players that are all people that are all online playing the game together. Furthermore, the actions are much different such as clicking the map to place routes and for better visualizations while mine requires the use of buttons.

Structural Plan

The finalized project will be organized in different functions. Most of the code is chunked together based on their use such as each mode for switching screens are grouped together. For example, the first couple of functions relate to the home screen and then the next few functions are the help screen etc. Much of the information and data is stored in the app parameter.

Algorithmic Plan

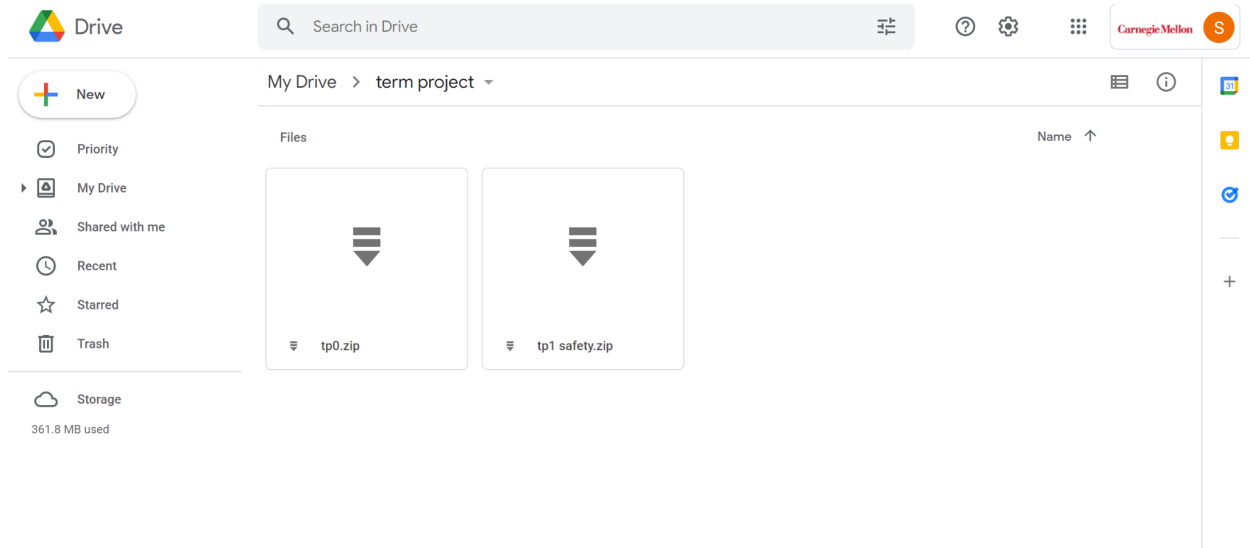
The part of my project that is algorithmically most complex is (hopefully) creating three game AIs against the human player. I will approach this using the Fundamentals of Game AI guidelines given by the 15-112 Notes. More specifically, I will model my algorithm similarly to the minimax pseudocode provided by the notes. The pseudocode would then be the basis for the other AIs that will be integrated into the project.

Timeline Plan

I intend to complete the rest of the graphics by this coming monday so that I am able to focus mostly on game AI which will hopefully be completed by tp2. Game AI will be a part of the MVP, therefore, it is imperative that I finish game AI by that time. If possible, I would also add sound effects for after when MVP is achieved.

Version Control Plan

I am backing up my code by storing all my zip files into my google drive term project folder. Each time I submit into Autolab, I will add the same zip file into the folder so that there are multiple versions of the code.



Module List

Not planning to use any additional modules.

TP2 Update

No Game AI is implemented rather it is two users on the same board. There is an added function of saving the game and then loading the game.

TP3 Update

Project Description

2048

This term project is 2048 inspired from the online game 2048. It is a Python simulation game that includes three modes: classic, impossible, and user-input versus game AI. The premise of the game is for the player to achieve 2048 on their grid.

Competitive Analysis

From the term project gallery, there are previous versions of 2048. My graphics are similar to the style of 2048 2.0 graphics. The initialization and the help screen are similar. Differences include: impossible mode which changes color and swaps key functions while 2048 2.0 uses easy, medium, and hard mode.

I believe that the graphics used by the previous projects were created by specific modules such as pygame or cmu_112_graphics. In my game, I use Turtle to create the game.

Structural Plan

The finalized project will be organized in different functions and modes. Most of the code is chunked together based on their use such as each mode for switching screens are grouped together. For example, the first couple of functions relate to the home screen and then the next few functions are the help screen etc. Much of the information and data is stored in the app parameter. They are also separated in different files based on the mode and essential parts of the game.

Algorithmic Plan

The part of my project that is algorithmically most complex is a game AI against the human player. I will approach this using the Fundamentals of Game AI guidelines given by the 15-112 Notes. More specifically, I will model my algorithm similarly to the minimax pseudocode provided by the notes.

Module List

Not planning to use any additional modules.