Project Name: Mini Metro
Project Description: I plan to create a 112 version of the game “Mini Metro”. The game entails creating a metro system by drawing lines between stations transporting passengers to their destination. As the game progresses, more stations appear along with more passengers to carry. However, each level, you unlock a new metro line allowing you to create more routes for the passengers to travel. The key idea of the game is to draw your lines in a strategic way to optimize travel.

Similar Projects:
This game has already been created and is quite popular. I have attached two links that display the premise of the game (https://dinopoloclub.com/games/mini-metro/) and gameplay (https://www.youtube.com/watch?v=E_d6CC-wNEU). I will be including the fundamentals of this already produced game but simplifying the mechanics and cutting out some additional features. The game begins with a handful of stations and 1 metro line to connect them. Passengers slowly spawn and the car on the metro line goes back and forth along the track delivering passengers to their destination. Their destination is determined by their shape, with each station having a corresponding shape that matches with passengers. Each passenger calculates the fastest path to their destination, as there are multiple stations to get off and multiple paths to take.The game is divided into “levels” of sorts which are determined by the number of passengers that you have delivered. During each level new stations appear along with passengers spawning at each station, as the player must strategically use your metro lines to efficiently deliver passengers before they become impatient. If a passenger remains at their starter station too long, you lose. The game is endless so each level has more passengers, more stations, and more tools to help you connect more stations and transport people more quickly. The number of passengers delivered is your total score which is the main goal.

I will be including the core mechanics of the game with the randomly generating stations, passengers spawning at each station, and the main tool connecting the stations with metro lines. I will definitely be using less smooth and flashy visuals and audio. Also, I will not be implementing unnecessary mechanics such as terrain that spawns that requires tunnels, a finite resource that you must spend wisely, to cross. At the end, if I have time I will attempt it. I will also include basic UI, but very bare bones, similar to tetris. I believe the fundamentals will be complex, difficult enough to implement.

Structural Plan:
I will create classes for each of the major object types in the game: Stations, Metro Lines, Passengers, Additional Upgrades/Obstacles. I have already begun coding this up. Each class will have a storage system for each object, with necessary attributes, and helper functions, methods for that class, to correctly maintain and update the game. Each will be very self-contained with just the animation class just accessing each one to change the visuals. For the visuals of the game, I will be sticking with 112 graphics and some image imports as the game is very visually simplistic. I will create a rectangular board using a similar one to Tetris for station placing. I will be writing many helper functions to update the visuals, update each object type, and then for game progression. All the code will be contained in one file.

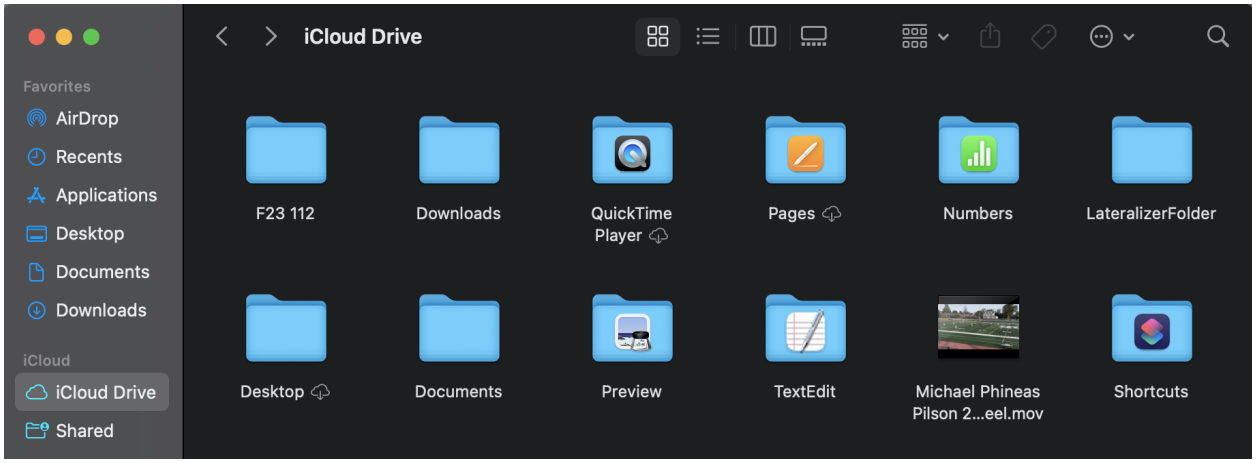
Algorithmic Plan:
The trickiest part of the project resides in the graph structure of the game and the pathfinding of the passengers. I will be creating a weighted, undirected graph structure for the metro network of stations and metro lines, with the stations being vertices and the metro lines being edges. I will use adjacency dictionaries which I am already comfortable with from individual research and the 112 TP special topics. I will be implementing my own version of path finding using Dijkstra's. This function will slowly build up a dictionary of the shortest path to each vertex based on the connections of the stations and the weights of the edges. The pathfinding will have to be run with each new passenger that spawns as that will affect the stops of the metros along their tracks and a potential speed change (I may have the metros run at a constant speed to simplify things). The weights will be a combination of the distance between the two vertices, the position of the metro along that track, and the possibility of switching tracks will have to be included. I have already discussed this with multiple TA's. I think I will store each metro line as a separate graph, and then combine them into one with additional vertices that represent switching metro lines at a specified station.

Timeline Plan:
By TP1
Graph structure
Start Path Finding

By TP2
Classes and Basic Mechanics
Finish Path Finding

By TP3
Visuals, Audio, UI

Version Control Plan:
I will be storing my progress in my iCloud cloud. I have already saved all my files to iCloud and they will update each time I edit them.



Modules: None

TP1 Updates:
I have made substantial progress in this first week. I have somewhat reversed the order in which I thought I was going to complete tasks. I have coded the structural foundation of the project. Established classes for the vertices, stations, passengers, and beginning the metro cars themselves. Currently, the user can easily connect stations to one another with different edges; however, I am struggling in how to implement the feature of deleting edges. I will ask my mentor but currently, I am thinking deleting each edge one by one, an easier way, but how would I deal with overlapping edges of different types? Which edge would be selected? The original game uses a “handle” of sorts to edit the edges which seems difficult but may be the best method. Also, I am debating on whether to implement images into the game. I think it will make the UI cleaner. Also, I will be starting to code the metro cars themselves. I will create their tracks by searching through the edges and following their path along with the graph. Then create another function that handles going between 2 vertices at a time. The pathfinding will be last. Everything else seems to be going to plan.

TP2 Updates:
I essentially have the game up and running. Players can easily and intuitively add and delete edges of each color. Metro cars automatically spawn on the first edge created in a class, they also despawn if there is only one edge left to delete. They correctly travel along their path reversing when necessary carrying passengers. Passengers enter the metro when it reaches the station. If their destination is on the route, they stay on. If it is not on their route, they get off at the next station that offers a new route. When they reach their destination, they get off and the score increases. The game is done. The things to improve are to create a title/settings screen to explain rules, maybe add music, and dial up the pathfinding.

TP3 Updates:
I have not added anything too significant to my project, more just polishing it. I have improved the pathfinding such that a passenger can make an educated decision on which to get off. If it is their stop, get off score increases, if it is not their stop but destination is en route stay on, if destination is not en route but the station has other lines available get off, and finally if destination not en route and stop does not have other lines stay on. The passengers always get on. I have added a title screen that includes a description of how to play at the beginning of the game. I have gone through and improved the style of my code, cutting out some magic numbers, including comments, and organizing the code into sections.