# Project Description:

Escape From CMU is a 2D single-player platforming video game that stimulates the process of escaping from CMU with the help of a secret organization.

# Competitive Analysis:

There’re three parts in my game: *Jump to Mellon Institute*, *Find Taylor’s Hat* (maze), and *Escape from CMU*.

The *Jump to Mellon Institute* part is similar to Doodle Jump. Doodle jump has more features including different platforms, different kinds of items placed on the platforms, and monsters. The player can move smoothly in the game. I refer to Doodle Jump a lot when designing the Jump to Mellon Institute part.

There’re many maze-solving games online, and the *Find Taylor’s Hat* maze is going to be similar to them, but it would be integrated with items in the main storyline and the player will get the chance to regenerate once when getting stuck.

When making Escape From CMU, I was inspired by a series of popular games that I’ve played so far. However, it’s going to be different since this is a CMU-oriented game that integrated different elements.

# Structural Plan:

The main file would include all the models (app.Started), controllers, view functions, and the life heart system based on different modes. The functions of each mode would be written in its own file. The major mode of the game would be (not including connecting screens):

* mainMenu mode
  + Display main menu
* *Jump to MI* mode
  + Initialization (init)
  + Class jump to MI
    - Platform generator
    - Player movement
    - Screen scroller
* *Find Taylor’s Hat* Maze mode
  + Class maze
    - Wall list
    - Solve for solution
* Escape Mode
  + Enemy class
    - Detect range
    - Movement
    - Chasing algorithm
  + Player movement
  + Win condition

# Algorithmic Plan:

In the three parts of my project, I plan to use the following algorithm:

* *Jump to Mellon Institute*
  + Infinite Terrian Generation
    - Randomly generated all the platforms when scrolling upwards.
    - (Challenging part) Decrease the amount of blocks when getting closer to the destination and make sure the blocks are not too far away from each other so the game can still be playable.
  + Physics of jumping
    - I created different stages of jumping, so the velocity is different at different locations. With acceleration involved, the player can bounce when colliding with platforms.
    - (Challenging part) Catch the collision when the player is falling fast and not exactly land on the platform when timerFired.
* Potentially add more features and monsters
  + Platform variation: jump and disappear / moving block
  + Platform items: springs / rockets
  + Monster: hit and die
  + Gun: shoot when mousePressed
  + Music

* Maze generation
  + Randomly generated solvabale maze (UPDATED)
    - Wall class + cell class
    - Using recursion and backtracking randomly generate a maze
    - Based on Depth-first Search Algorithm
    - Remove all the walls in between to create a path
  + Solve maze and display solution for ten seconds as a hint (UPDATED)
    - Solve maze
    - Display when player use a hint
* Back to campus
  + Read the map and detect legal move
    - Use OpenCV to get the color of each pixel as an RGB string and compare that to the path color to see whether it’s a legal move or not.
    - The map would move relative to the player’s location.
  + Enemies involved
    - Every enemy has a certain detect range (show as yellow light) and a different walking path.
    - When inside the detect range, the enemy uses Dijkstra's algorithm to find the shortest path and chase the player. Enemies have different chasing speeds.

# Timeline Plan:

Done by TP1:

* Nov. 18 Jump to MI basic features (DONE)

Done by TP2:

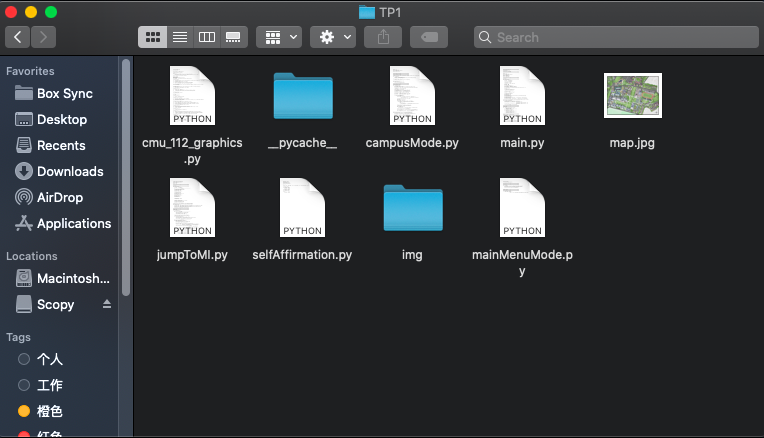
* Nov. 20 Maze generation part (DONE)
* Nov. 22 Different screens connected (including win page) (DONE)
* Nov. 23 Campus Map part: player able to move and reach the win condition (DONE)

Done by TP3:

* Nov. 25 Enemies introduced
* Nov. 27 Enemies able to chase and track
* Nov. 30 Feature added to jump to MI
* Nov. 30 UI improved (all stories presented)
* Dec. 1 All extra features added
* Dec. 2 Video

# Version Control Plan

I backed up all of my files using Box Sync.



# Module List

OpenCV

Numpy

Pygame