**Project Proposal**

**Project Description**: The name of the term project and a short description of what it will be.

* Time Locker: 112 Edition
* The project will be a 2D shooter with a simplistic pixelated artstyle where the player aims to get a high score. The player can get points by moving forward and killing enemies. The game will have an endless game map with randomly generated obstacles and enemies. If an enemy or a projectile they shoot touches the player, the player loses. There will be a variety of enemies with differing amount of health, move speed, and projectiles. These enemies will spawn in from the forward, right, or left direction and will walk in a straight line. There will also be special enemies that track the player and follow them. The player controls their character’s movements using the arrow keys. The character only shoots projectiles when it is moving. In the game, time slows if the user moves slowly, and time will fast forward if the user moves quickly. The speed of the flow of time will dictate how fast the enemies move. To move quicker, the player can hold down an arrow key which will speed the movement up in that direction up to a certain maximum speed. To move slower, the player can tap on the arrow keys to move a little bit at a time. If the player is not moving at all, time pauses and the enemy freezes. Additionally, there will be a “shadow” that will start behind the user and move forward, engulfing the map. If the player is caught by this shadow, they lose, giving the player the incentive to keep moving. There will also be items that the players can pick up which give it special additional projectiles. These items will be dropped by special enemies that randomly spawn.

**Similar projects**: A 1-2 paragraph analysis of similar projects you've seen online, and how your project will be similar or different to those.

* Time Locker: This is the 2D shooter mobile game that my term project took most of its inspirations from. It is an indie game made in the unity game engine. The mechanics of my game are all directly taken from this game, so they are very similar. There is a difference in controlling the movement of the character since this is a mobile game, the player moves around by swiping on the screen. They can control time through the speed at which they swipe it. One mechanic of the game I did not incorporate was that in this game, characters are color coded based on their types (normal, drop items, drop money, follow player). This is because I am not sure what exactly I will use for the artwork for enemies yet. I might choose to distinguish their types by their actual design and other aspects of them (like speed and health bar) by their size. Also, the game has a currency system, where the player can use the currency they collect to buy new temporary characters with special projectiles after every round. The players unlock more of these temporary characters the more they play the game. This progression system is something I will not implement. There are a variety of special projectiles the players can pick up in the game, and although this is a feature I plan to implement, it is unlikely I will add all of them to my game.
* SUPERHOT: This is the game that Time Locker takes inspiration from. It is a first-person shooter game that employs a similar time-manipulation mechanic. Although the time-manipulation mechanic is largely the same as the one present in my game, all other aspects are quite different. The game is 3-dimensional and involves the player traversing through a room and eliminating enemies through a variety of methods. It is much more complex than both my project and Time Locker. The game has a very simplistic yet stylized art style which Time Locker attempted to replicate on a smaller scale. My game will also take this approach to visual design as it places the unique mechanic of the game on the forefront.

**Structural Plan**: A structural plan for how the finalized project will be organized in different functions, files and/or classes.

* Functions:
  + onAppStart(): initialized all the global variables that I will need (game speed which can be expressed as stepsePerSecond, positions of the enemies, positions of the projectiles, paused game state, player score, etc).
  + redrawAll():
    - Will draw the player’s character in a fixed a position on the canvas every time. As the player “moves” using the arrow key, the character should stay still and the enemies around it and the obstacles should move.
    - Have a function in a for loop that draws each of the enemies
    - Similarly, have another function that draws the projectiles (and another 2D list that tracks that).
    - Have a function draw the shadow that moves behind the player
  + onKeyHold():
    - if the user moves one of the arrow keys: move them towards that direction (which actually means moving everything else towards the other direction) and increase the counter for the game speed
  + onKeyRelease():
    - when arrow keys are released (none of them are pressed), reset the speed of the game to 0
  + onKeyPress():
    - ‘p’ should pause the game
    - ‘r’ should restart the game
  + onStep():
    - Take steps which allows everything to move
    - Have a function that randomly add new enemies to a 2D list of the position of enemies (which should be a global variable).
    - Have a function that checks if anything (enemies, projectiles, shadow) overlaps with the player. If they do, then kill player and end the game.
* Classes:
  + Player: This class will be for the player character. It will keep track of its size and the projectiles that it shoots.
  + Enemies: This class will be for all the enemies in the game. An object of this class will have the property of health, size, move speed (a multiplier to the game speed), and image (which allows artwork to be imported for these enemies, so I do not have to create my own), and their type (whether they drop items or follow the player), and direction (which dictates the direction they move in). Obstacles would also most likely be belonging to this class but instead they will have a 0 move speed and be placed on the middle of the map instead of spawning on the sides.
  + Projectiles: This class will keep track of all the projectiles which are from either the enemy or the player. It will keep track of properties such as speed and size as well as duration between shots.
  + App: This is the class already built into CMU graphics. This will be the class I use to keep track of global variables such as the speed of the enemies and projectiles (which will be modified as the player is moving) as well as the score of the player.

**Algorithmic Plan**: A plan for how you will approach the trickiest part of the project. Be sure to clearly highlight which part(s) of your project are algorithmically most difficult, and include some details of how you expect to implement these features.

* Random generation of enemies and obstacles: I will approach this part of the project by writing a function that deals with the random generation. The number of enemies that are generated should be random and this could be done by importing the random module. The type of enemies can also be generated using random in a similar way since we can simply put all the objects of the enemy class into a single 2D list. The locations of the enemies or obstacles can be kept track of in a 2D list and can be modified to make them move. Enemies that go offscreen should be removed from the list and new ones can be generated.
* Time manipulation: This will be done using the onStep() built-in function and the app property stepsPerSecond. The onStep() function will continuously take steps to update the positons of the enemies (changing the speed at which they move at). The stepsPerSecond variable will be updated based on the player’s key holds and releases which then dictates how often the onStep() function is called and how fast everything moves.

**Timeline Plan** [2.5 pts]: A timeline for when you intend to complete the major features of the project.

* After TP0: Begin working on code.
  + Implement basic framework of the project. Write global variables and write define all functions (don’t need to fill them out, leave them blank and write “pass”). Write the function that draws the player character (can be geometric shapes) and the player class. Write the enemy class and try to spawn in enemies. Write all functions in onKeyHold, onKeyRelease, and onKeyPress. Write onStep() for the time mechanic. Implement mechanic that detects whether the player has been hit.
* After TP1:
  + Focus on random generation of enemies and obstacles. Implement features that I did not get to from before TP1. Write projectile class that allows the addition of different projectiles. Implement special enemies that track player or drop items.
* After TP2:
  + Implement external artwork (images) into the game to make it look nicer. Create UI elements (a nicer pause screen, a tutorial, etc). Debug existing code. Improve upon existing features or add new ones if there is time leftover.

**Version Control Plan**: A short description **and image** demonstrating how you are using version control to back up your code.

* Version control will be done through GitHub. I downloaded Git onto my computer and linked with my GitHub account on VSCode. I then initialized a Git repository based on my Term Project folder.

**Module List** [1 pts]: A list of all external modules/hardware/technologies you are planning to use in your project. Note that any such modules must be approved by a tech demo. If you are not planning to use any additional modules, that's okay, just say so!

* **Storyboard** [5 pts]  
  Generate a storyboard that demonstrates how a user would interact with your finished project. A storyboard is just a series of sketches showing (roughly) what your project will look like. Your storyboard should have at least six panels, and at least three of those should demonstrate features within the project. You may scan or take a picture of your storyboard and include it in the directory as the file storyboard.png (other acceptable filetypes include .gif, .jpg, and .pdf). Note you may not use screenshots or images from existing applications or projects. You should create these from scratch. However, you can take inspiration from existing applications when designing your own.