Final Project

Objectives

- Create a full game from scratch using either LÖVE or Unity.

From Start to Finish

It's time to begin the course's culmination: your final project! While most of the course thus far has been a series of assignments centered around adding features to existing code bases, this project will take place from the first line of code to the last, a true end-to-end experience to help tie everything we've learned together thus far. The following considerations should be met while designing and implementing your project:

- 1) Your game must be in either LÖVE or Unity.
- 2) Your game must be a cohesive start-to-finish experience for the user; the game should boot up, allow the user to play toward some end goal, and feature a means of quitting the game.
- 3) Your game should have at least three `GameState`s to separate the flow of your game's user experience, even if it's as simple as a `StartState`, a `PlayState`, and an `EndState`, though you're encouraged to implement more as needed to suit a more robust game experience (e.g., a fantasy game with a `MenuState` or even a separate `CombatState`).
- 4) Your game can be most any genre you'd like, though there needs to be a definitive way of winning (or at least scoring indefinitely) and losing the game, be it against the computer or another player. This can take many forms; some loss conditions could be running out of time in a puzzle game, being slain by monsters in an RPG, and so on, while some winning conditions may be defeating a final boss in an RPG, making it to the end of a series of levels in a platformer, and tallying score in a puzzle game until it becomes impossible to do more.
- 5) You are allowed to use libraries and assets in either game development environment, but the bulk of your game's logic must be handwritten (i.e., putting together an RPG in Unity while using a UI library would be considered valid, assuming the bulk of the game logic is also not implemented in a library, but recycling a near-complete game prototype from Unity's asset store with slightly changed labels, materials, etc. would not be acceptable).

How to Submit

1. If you haven't done so already, visit this link, log in with your GitHub account, and click **Authorize cs50**. Then, check the box indicating that you'd like to grant course staff access to your submissions, and click **Join course**.

2. Using Git, push your work to `https://github.com/me50/USERNAME.git`, where `USERNAME` is your GitHub username, on a branch called `games50/assignments/2020/x/final` or, if you've installed `submit50`, execute

   ```
   submit50 games50/assignments/2020/x/final
   ```

   instead.

3. Record a 1- to 5-minute screencast in which you demonstrate your app's functionality and/or walk viewers through your code. Upload that video to YouTube (as unlisted or public, but not private) or somewhere else.
4. Submit this form.

You can then go to https://cs50.me/cs50g to view your current progress!