Project Proposal: Simile (Spelt correctly)

Simile is a pixel-art roguelike, action-adventure video game developed using Pygame. It will be released for Windows, Linux, and Mac OS. The player can choose between two different characters, Smile and Sly, each with their own unique skillset. The characters can slice, shoot, and dodge stuff.

The world, called Spirit, has different locations (rooms). Spirit is filled with dangerous enemies, all waiting to kill you. As you traverse through the bewildering world of Smile, you will encounter NPCs, who will guide you in your journey to a gun with the ability to destroy your past.

Competitive analysis:

There's a coding video on YouTube titled "Creating a Zelda style game in Python [with some Dark Souls elements]" by Clear Code which is similar to the game I'm creating. The logic I use for spirit groups and camera movement will be similar to the way he implements it. However, shooting, dodging, enemies' AI, NPCs' AI, the world, and a lot of other stuff will be completely different.

I'll also be using his tutorials to guide myself when creating the game.

Here's the link to the video: https://www.youtube.com/watch?v=QU1pPzEGrqw&t=13764s

Structural Plan:

The game file is divided into two folders:

- source code – contains all the .py files
- graphics – contains all the images, textures used for the game

Each element in the game world will have their own class. So far we have the following class:

- Game
- Level
- Player
- Tile

Algorithmic Plan:

The trickiest part of my code will be animating, moving, and managing all of the enemies, obstacles and the player at the same time, while also managing the interaction between the enemies and NPCs.

To tackle this, I'll create two sprite groups, visible sprite and obstacle sprite. All of the elements that's visible on the screen will be put inside the visible sprite group, and all of the elements that interacts with the player and NPCs will be in obstacle sprite. Visible sprite group will be used to animate and display all the elements and obstacle sprite group will be used to manage all the interactions.

Timeline Plan:

Oct 23: Develop the world, implement player movements.

Oct 27: Create animations and particle effects for player and enemies.

Oct 29: Implement Enemies AI. Fill the world with enemies. (MVP)

Nov 1: Submit TP2

Nov 5: Implement NPC AI. Create Story.

Nov 7: Submit TP3

Version Control Plan:

Git and GitHub

Module List:

Pygame and other general python libraries like random, date, etc.