CS324E Final Project Plan

Team name: GenerateRandom()
Project group members: Josh, Haris Bhatti, Marilia Sampaio
Project name: GeneratePortal<?>

FInal project proposal instructions:

The initial project plan will break down the scope of the project and its intended functionality, summarizing required features and any additional bonus features beyond the baseline. This report will describe project challenges and breakdown of work per groupmate.

Project description:

Our game will be similar to the all time classic game "portal." We will have a player equipped with a portal gun which they will use to traverse each level to escape through an exit door. Throughout the level there will be certain obstacles such as "lava" or "spikes" which the player cannot touch or else they lose a life or the game ends. The map data will be generated by reading an XML or JSON file that describes a scene and the spawn point of the player. There will be about 3 or 4 levels with progresive difficulty.

- Features:
    - 3 lives
    - Portal transportation of the player
    - Obstacles (walls, spikes)
    - Sprite animations - jumping, shooting (where user is aiming with mouse input)
    - Projectile mapping (in/out portal which the player shoots)
    - Gravity applied to the player
    - Level complete timer
- GUI systems/pages:
    - Main menu page
        - Buttons and display information
        - Brief description of objective (escape the level)
    - Game page
    - Game over page
    - High score display
    - Lives display
    - Pause display
        - Pause button
    - Restart display
        - Restart keypress
    - Level display
- Class objects with animation hierarchies
    - Class 1: Timer
        - Attributes: int start_time, int stop_time, int time_elapsed, boolean running, boolean resume

        - Methods: void start, void resume, void stop, int getElapsedTime (returns elapsed time), int second (computes seconds), int minute (computes minute)
    - Class 2: GUI
        - Attributes: PImage life;
        - PImage victory;
        - PImage defeat;
        - PImage menu;
        - Boolean displayBanner;
        - Methods:
            - Void level, void, lives, void displayLossBanner, void displayWonBanner, void displayScore, void StartMenu, void PauseButton
        - Constructor: void GUI()
    - Class 3: Player
        - Class 4: portal gun (connected to the player)
            - Attributes: player_shoot, direction, velocity, hit_wall
    - Class 5: Obstacle (wall)
        - Attributes: boolean portal_on_wall
    - Class 6: Gameboard
        - Mainly used to restart/redisplay game
        - Inherit Obstacle class, GUI class
        - Constructor: components inherited from GUI class, either PImage,
        - Attributes: boolean won_game, int num_lives,
        - Methods: void display
- Data Input/Output
    - User created data to provide settings
        - Respawn coordinates from user input/shooting portal
        - Map builder to generate level map based on XML/txt file
        - Data saved from application (high score and time)
    - User buttons/keys
        - Pause the game
        - Move the sprite (jump, right, left)
- Sound
    - Shooting sound when you shoot a portal (must be mutable)

Project Challenges:

- Game logic
- Sprite animation/respawning
- Sprite positioning
- Use of physics on sprite and applying forces that occured before entering the portal

- Connecting response from other classes to GUI
- Map data read from a XML/JSON file

Work division:
Josh - Game logic Including: player interactions with the environment, projectile mapping, portal placement, file input, etc.

Marilia - Game logic including: movement, class construction, sprite interactions, portal placement, etc.

Haris - GUI class and how to implement it and connect it to the sprite and user inputs. PImage's that will display when the state if the game changes (start screen, game over screen, high score screen).