AP Computer Science Final Project - README Template

Instructions:
The first step in creating an excellent APCS final project is to write up a README. At this stage, this README file acts as your **project proposal**. Once you've filled in all components, Shelby will read through it and suggest edits. Ultimately, you need a document that adequately describes your project idea and **we must agree on this plan**.

Have one member of your group **make a copy of this Google Doc**. Then, they should share it with all other members **and with Mr. Shelby** so that every group member has edit permissions, and Shelby can add comments on your ideas.

There's a lot of parts of this document that you might not have full answers for yet. Because you haven't written the program yet, it's difficult to think about the **instructions** or **which group members will do which parts**. Even though this is hard to think about, you must have something in these sections that acts as your current plan. However, during the course of the project, you'll **continuously update this document**. This means that you will not be *held* to exactly what you put here - components of this document can change (and it's pretty common!).

There is one exception: the **Features List** section. Once Shelby OKs your README, the Features List section **cannot be modified**. For this reason, it is most important that you get a solid idea of what you want to make and the primary features it will have *now*.

Talk with your group. Consider drawing some pictures of what you think your project might look like. Be precise. When you're ready, fill this out together. Each component in brackets below ([these things]) should be replaced with your ideas. Note that there are several sample READMEs posted on this assignment for you to use as guidance.

-------------------When README is finalized, remove everything above this line--------------------

[Void Jump]

Authors: Jonathan Levi, Ilai Tamari, Martin Simeonov
Revision: 5/13/2022

<u>**Introduction**</u>:
The game is based around a futuristic science experiment that didn't go well. Robert the scientist was on the verge of discovering time travel, but as soon as he clicks the button he realizes his machine functions in an absolutely different way.

Clicking the button made Robert lose consciousness and after a few hours he wakes up in a room with only a gun next to him. When he picks up the gun and shoots at the nearest wall, he realizes this is not a normal gun - it's a portal gun.
Robert, of course, wants to get out of this room and return to his lab, so he uses the portal to get to the exit. However, when he reaches the door, he finds out this is only the beginning and that there are many more levels to complete before he can figure out what has happened.
Help Robert return to his lab by moving from platform to platform and controlling his movement and his portal gun to get to the door. Sounds easy? Always keep in mind that platforms might be unexpectable as they might affect your movement or in some cases kill you.
This game involves thinking ahead, planning strategies, and moving fast, and the notes found in the beginning of each level will keep you curious about what is going on and why Robert forgot how he got there.
The idea is based on the video game Portal, but it includes other features such as platforms, special abilities, and a developing background story.
How to play? When a level starts, you will find Robert standing on a platform on the left side of the screen, and your goal is to help him get to the gate on the other side. You can help him by shooting portals on the screen (use right and left mouse buttons), moving right and left using left and right arrows, and jumping with the space button. Finish a level by simply standing in front of the door, and you will immediately be bright back to the menu where you can choose the next level.

Instructions:
When running the game, the user will be prompted to select a level and be spawned into the respective map. In order to play the game, the user will use the w, a, s, d keys (up, down, left, right) for movement as well as both the mouse buttons (left click and right click) for using the portal gun the player has to create portals.

Features List (THE ONLY SECTION THAT CANNOT CHANGE LATER):
Must-have Features:
- Portal Gun:
    - Player character is equipped with a portal gun, which the player can (and should) use to traverse their environment to reach the end. The player will be able to shoot out two portals at a time, and when they walk through one end of the portal, they'll end up on the other side. The portal, when shot out, will attempt to reach the position that the cursor is on. If its path is obstructed by an object, the portal, when it hits the object, will stop exactly where it collides with the object and stay there. Portals are vertical to the ground.
- Winnable:
    - The game is divided into levels, where in each level the player starts on a platform on the left side of the screen and their goal is to reach the right end of the screen, where there is a gate. The gate can only be opened if the player first touches a key that is also on the screen in a place where the player can reach it. The game is winnable, which means that when approaching the gate with a key, the level is done and the user is back in the menu.

- Player must be able to move
    - The player will be able to move left and right with the help of the arrow keys and will be able to jump using the up arrow key. However, the player will not be able to jump very high and will need to use the portals to complete the levels.
- Physics:
    - Only the characters are affected by gravity. If the player moves to a spot with no ground, they should fall and accelerate downwards (to a certain speed) until they hit the ground or hitting spikes, the latter resulting in a game over and a restart of the level. Furthermore, if the player goes through a portal, they should come out of the portal while maintaining the momentum they built up before entering the portal. Platforms hung in the air shouldn't fall down.
- Multi-leveled game:
    - The game will have several different levels, with different maps and obstacles that the player must beat in order to win. There will be simpler levels, as a tutorial for the user (introducing them to the game), and then more advanced levels with various other obstacles.

Want-to-have Features:

- Monsters
    - Monsters will make it harder for the character to get to the gate. Some examples of monsters a character might face are ones that move right and left on platforms and ones that shoot short laser beams into the air. If a character touches a monster or its laser beams, the level restarts and the character goes back to their initial location on the screen.
- Different types of platforms
    - There will be different types of platforms that will act in different ways when stepped on. These platforms include: falling platforms (platforms that fall when stepped on), fire platforms (platforms that burn the player and kill him), size-changing platforms (platforms that become smaller and larger as time passes), speed-enhancing platforms (platforms that boosts the players speed and momentum when stepped on), and moving platforms (platforms that move in specific directions on the map).
- Stopwatch feature
    - A clock-like counter will be implemented in the game so that the player will be able to check how long it took him to finish the level. Moreover, the player will be able to challenge himself and try to get a new time record for finishing the level.
- Make portals gifs
    - When a portal is shot, a moving gif of a portal will be drawn on the screen causing the game to look more realistic and less computerlike.
- Notes that explain the background of the story + Robot voice
    - Every time a new level starts, a small note will appear on the screen. Each note tells a bit of information about the background story, but it is not absolutely clear until reading the last note. Also, a robot voice will introduce every level by greeting the player and reading the note out loud.

Stretch Features:

- Add multiplayer
  - The player can join up with other players to solve the levels together. More difficult levels with multiple gates will be added as well. Additionally, the multiplayer mode will contain cooperation levels as well as versus levels where the players will race each other to the gate.
- Animation
  - Includes moving animation, aiming animation, and portal explosion animation. Moving animation means making the legs of the characters move when the character walks, aiming animation means making the gun point up and down, and portal explosion animation means showing the portal going out of the portal to where it was aimed.
- Hard mode
  - Portals have time limits, causing them to explode when reaching it. Have an aging limit so that the player can't jump or else he would die. Additionally, there are only a specific amount of portals you can shoot in one level.

<u>Class List</u>:

- Person Class: has all fields and methods a character should have in the game
  - Player Class: constructs a character with a specific design and the ability to take keys
  - Monster Class: a character with limited abilities
- DrawingSurface Class: responsible for the flow of the game
- The Platform Classes:
  - Platform Class: A regular platform that allows the player and other characters to stand on it. The following classes extend this class:
    - Spikes Class: Much like the Platform class, except that it removes any character that hits it, essentially 'killing' that character. If the player hits the Spikes, the level will restart
    - Wall Class: Just like the Platform class, but vertical. Mostly used to block the player in some way. Furthermore, it blocks portals as well.
    - ForceBarrier Class: The ForceBarrier blocks portals from being placed beyond it, like the Wall, but the player can go through the ForceBarrier
- Lazer Class: Can 'kill' characters just like the Spikes, but it doesn't need to be attached to the ground or a platform
- Portal Class: defines what a portal is with all its methods
- Abstract Screen Class: An abstract class containing all fields and methods a screen should have
  - Beginning Class: The opening screen which first greets the player
  - Menu Class: The screen after the Beginning screen, where the player can choose a level to attempt
  - Level Class: The screen which displays the level the player has chosen from the Menu screen

- ScreenSwitcher Class: switches the screens when necessary

Credits:

Responsibility List

- Jonathan: All the Platform classes + levels brainstorm and implementation
- Martin: All the Character classes + levels brainstorm and implementation
- Ilai: DrawingSurface, Portal, animations and sound + levels brainstorm and implementation