AP Computer Science Final Project - README Template

Hypergunner

Authors: Daniel I'm, Edward Lu, Dhruv Sharma
Revision: 4/12

Introduction:
The project will be a rhythm game in the style of galaga with a player controlled ship at the bottom of the screen that can move left and right between various “lanes” to the beat of a song to avoid oncoming debris and move through a specific path (basically, there will be lanes that you will need to switch between for you to not crash based on the rhythm). Players can also press a button when in “energy fields” to shoot and destroy rocks in front of them at specific points in a song. The ship will have a set amount of health that decreases each time the player hits something(walls, debris, ectr). The map will scroll down past the player, and they will remain vertically stationary at the bottom to leave the most amount of space above the screen so they have time to react to oncoming obstacles. Maps can be stored as 2D arrays read from text files, and will likely have 3 or 5 lanes. Maps should ideally scroll quite quickly, though a balance between speed and playability will need to be considered, as moving too quickly for the sake of energy and rhythm could make the game impossibly hard. There isn’t much of a story, but the goal of the program is to complete all the levels, possibly with a scoring system, either numerically or with an D/C/B/A/S scoring system. Fans of games like Thumper will probably enjoy this, as their rhythm-based gameplay is somewhat similar.

Instructions:

Use the A and D or arrow keys to move lanes left and right, and the spacebar to shoot (there should be a delay so even if you do hold it down, there would be some instances where it does not hit anything), primarily keyboard controlled
A level select menu will be needed, and maybe another one at a higher level to decide between playing or level editing if that ends up being added

Features List (THE ONLY SECTION THAT CANNOT CHANGE LATER):
Must-have Features:
[These are features that we agree you will *definitely* have by the project due date. A good final project would have all of these completed. At least 5 are required. Each feature should be fully described (at least a few full sentences for each)]

- Shooting/destroying and collision mechanics
- Load and play music files, scroll levels at the right speed

- User key read for movement
- Loading Images/Sprites
- Tutorial/Guide

Want-to-have Features:
[These are features that you would like to have by the project due date, but you're unsure whether you'll hit all of them. A good final project would have perhaps half of these completed. At least 5 are required. Again, fully describe each.]

- Custom maps. Map editor or drag in a file
- Special Effect animations for perfect moves (something like not getting hit once)
- Scores/Scoreboard
- Custom sprites(for user ships)
- More levels or bosses and other things
- Powerups?

Stretch Features:
[These are features that we agree a *fully complete version of this program would have, but that you probably will not have time to implement*. A good final project does not necessarily need to have any of these completed at all. At least 3 are required. Again, fully describe each.]

- Super Smooth animation/movement
- Multiplayer(local)
- Custom music

Class List:
[This section lists the Java classes that make up the program and very briefly describes what each represents. It's totally fine to put this section in list format and not to use full sentences.]
Ship - represents the player ship
Bullet - represents the laser bullets players can shoot
Debris - represents destructible debris
Wall - represents indestructible walls

Credits:
[Gives credit for project components. This includes both internal credit (your group members) and external credit (other people, websites, libraries). To do this:

- List the group members and describe how each member contributed to the completion of the final program. This could be classes written, art assets created,

leadership/organizational skills exercises, or other tasks. Initially, this is *how you plan on splitting the work*.

- Give credit to all outside resources used. This includes downloaded images or sounds, external java libraries, parent/tutor/student coding help, etc.]