**Pacman Game**

[insert name]

CSC-17B-[class code]

April 22, 2019

|  |  |
| --- | --- |
| **Member** | **Role** |
| Benjamin Kwon | Ghost behavior |
| James Hartley | Game engine, game mechanic |
| Sebastian Hall | Login/Signup page, front end graphic |
| Ghislain Muberwa | High score page |
| Samuel J Hernandez | Database |
| Christopher Schlenker | Database connection between server and front end |

**1. Introduction**

Pacman is an arcade game and is regarded as one of the most influential video games of all time. For this project, the group decided to try and create an accurate representation of the original Pacman. The project was created using the Phaser game framework, which allows fast rendering of HTML5 games.

**2. Description/Gameplay**

As an arcade game, the gameplay and rules are relatively simple. The basic controls make use of each directional arrow key on a standard keyboard.

*Basic Controls*

|  |  |
| --- | --- |
| **Key** | **Action** |
| Up arrow | Move up |
| Down arrow | Move down |
| Right arrow | Move right |
| Left arrow | Move left |

The player, playing as Pacman, finds him or herself in an enclosed maze containing four ghosts in total. Each of these ghosts are programmed to chase Pacman with their own unique method. The details of these methods are explained more in the following table:

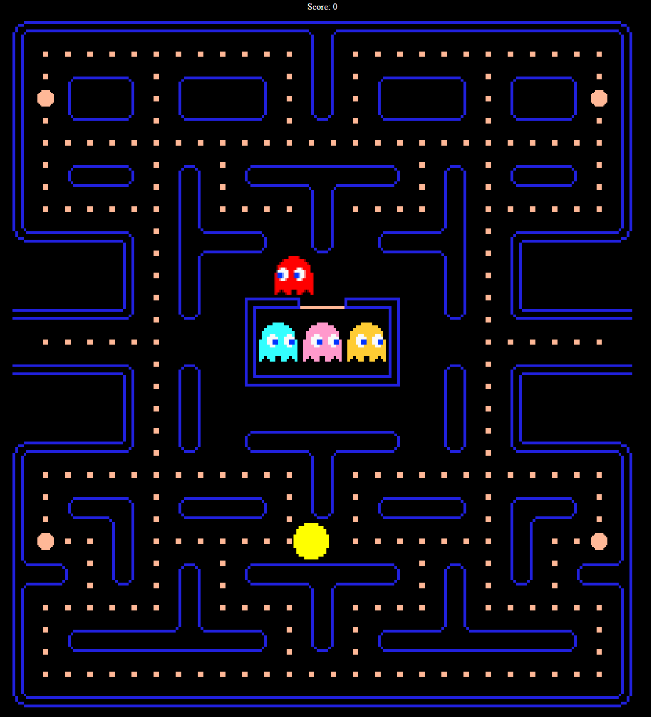
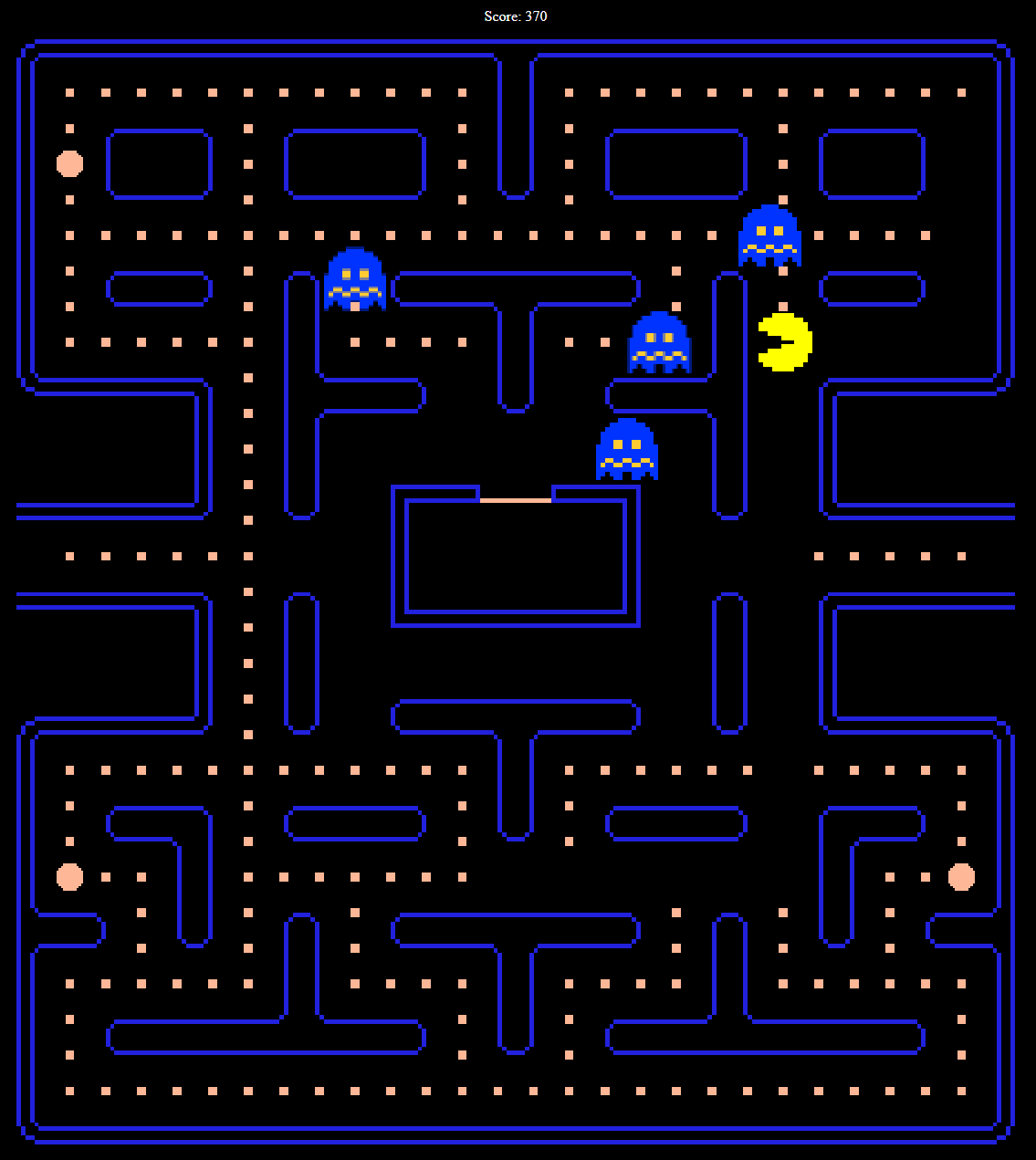
Table 1. Ghosts and their target tile (pathfinding algorithm)

|  |  |  |
| --- | --- | --- |
| **Ghost name** | **Color of ghost** | **Target tile** |
| Blinky | Red | Tile Pacman is on |
| Pinky | Pink | 4 tiles in front of Pacman |
| Inky | Teal | 1. Get position 2 tiles in front of Pacman  2. Double distance Blinky is away from (1)  The resulting vector end point is target tile |
| Clyde | Orange | If more than 8 tiles away from Pacman,  target tile is identical to Blinky’s  If less than 8 tiles away from Pacman,  flees to bottom left portion of map |

The maze contains tiny dots called pellets as well four larger dots called power pills. The goal of the player is to collect every pellet without being caught by a ghost. If Pacman collects a power pill, Pacman becomes invincible and will not die if contact is made with a ghost. Instead, it will eat the ghost, which will run away back to the center of the map and revive a few seconds later. While Pacman is invincible, the color of every ghost will turn blue, to signal their vulnerability.

To goal of the game is the obtain the highest score possible. Eating pellets, power pills, and ghosts all give points in increasing order. The player must try to collect every pellet on the map without being caught by a ghost. If caught, the player will lose a life. The total lives or chances available is three.

To play the game, users must register an account and log in. Once logged in, they will be granted access to the game. When the player loses all his lives and the game is over, the score will be sent to the database and be connected with the player’s username. If the score is within the top ten current scores, it will be shown on the webpage hosting the game.

*A capture of the starting gameplay* *A capture showing the effects after eating a power pill*

**3. Concepts**

Code structure/Objects/Model-View-Controller

The structure of the code is divided up into distinct sections. When the game runs, it is initialized, preloaded, and then created.

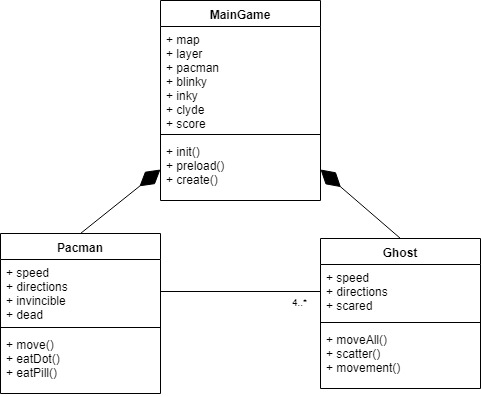
In the initialize function, the libraries and graphics that run the game are scaled and rendered. In the preload function, all the assets of the game including images, sprites, tilemaps, and audio are loaded. Lastly, in the create function, using the loaded assets, all the characters and map data are constructed and placed. Here, Pacman and the ghosts are constructed objects of their respective class. This can be seen in the MainGame.js file.

The main structure of the game engine is the update loop, which continuously calls on the necessary functions required to keep up with a real-time game. Here, the positions of each sprite are tracked, and the movement of each sprite is determined by the current state of the game. For example, the keys (controller) pressed by the player are constantly being checked and sent to the model, which determines the path taken by the player. This is then shown through the view, which is what the user sees while playing the game. The update loops can be seen in Pacman.js and Ghost.js files.

All the ghost’s behavior will be implemented; their initial state of how they are supposed to interact with Pacman and how they behave after a power pellet is eaten. The ability to defeat the ghosts by eating a power pellet needs to be implemented so that Pacman can eat them instead of going through them and have the death animation for the ghost occur.

Database needs to be created to store users, passwords, and scores. The player should login before starting to play the game so that their score will be tracked. After a game session is over, the score should be saved to the server. A front page will be implemented for the login/sign up screen for the player and another page to display a list of scores for the login player, and a play button that leads to the game.**5. Diagrams**

UML Class Diagram



Flowchart

A close up of text on a white background

Description automatically generated

**6. Code**

The source files for the project can be found at the group’s GitHub repository at

<https://github.com/SebastianHall69/PacmanGroup2>.

A web server will be needed to properly play the game on a browser.