Jacob Stewart

COP-2939-86403

Instructor: Dr. Banisakher

Date: March 10, 2024

Preliminary Design

CAPSTONE: PACMAN IN JAVA

Table of Contents

[1. Overview 2](#_Toc160964239)

[1.1 Overall Structure 2](#_Toc160964240)

[1.1.1 Game Loop: 2](#_Toc160964241)

[1.1.2 Game State: 2](#_Toc160964242)

[1.1.3 Entities: 3](#_Toc160964243)

[1.1.4 Collision Detection: 3](#_Toc160964244)

[1.1.5 Input Handling: 3](#_Toc160964245)

[1.1.6 Rendering: 3](#_Toc160964246)

[1.1.7 Map Design: 3](#_Toc160964247)

[1.1.8 Game Rules: 3](#_Toc160964248)

[1.1.9 Score Keeping: 3](#_Toc160964249)

[1.1.10 Sound Effects: 4](#_Toc160964250)

[1 Frameworks and Software 4](#_Toc160964251)

[2.1 Frameworks 4](#_Toc160964252)

[2.2 Software 4](#_Toc160964253)

# Overview

The purpose of this document is to present the preliminary design for pacman. The systems architecture revolves around its core game loop that is modified using a basic state machine to record which section of the game is currently in play (i.e menu, game, death, loading). Each entity in the game is responsible for its update and render loops when is then passed into the core game loop.

## Overall Structure

### 1.1.1 Game Loop:

The game loop controls the flow of the game, updating the game state and rendering it to the screen. It consists of a loop that repeatedly performs the following steps:

* Process user input.
* Update the game state based on the input and current game logic.
* Render the game state to the screen.

### 1.1.2 Game State:

Represents the current state of the game, including the positions and states of all entities (Pac-Man, ghosts, pellets, etc.). It also includes the current score, level, and lives.

### 1.1.3 Entities:

Each entity in the game (Pac-Man, ghosts, pellets, etc.) is represented by a class. Entities have properties such as position, direction, and state, as well as methods for updating their state and rendering them to the screen. Entities also have collision detection logic to detect collisions with other entities.

### 1.1.4 Collision Detection:

Handles collision detection between entities, such as between Pac-Man and a ghost, or between Pac-Man and a pellet. Uses rectangle objects to create algorithms to determine when entities collide notably using Rectangles built in Intersect method.

### Input Handling:

Captures user input, such as keypresses, to control Pac-Man's movements. Converts user input into commands that update the game state (e.g., moving Pac-Man up, down, left, or right).

### 1.1.6 Rendering:

Responsible for rendering the game graphics to the screen. Uses a rendering library (e.g., JavaFX, Swing) to draw the game state (entities, maze, score) to the screen.

### 1.1.7 Map Design:

Represents the game levels, including the layout of the maze, the placement of pellets and power pellets, and the starting positions of Pac-Man and ghosts. Uses data structures (e.g., arrays, lists) to represent the maze layout recorded from a text file.

### 1.1.8 Game Rules:

Enforces the game rules, such as what happens when Pac-Man eats a pellet or collides with a ghost. Controls the game progression, such as advancing to the next level when all pellets are eaten.

### 1.1.9 Score Keeping:

Tracks the player's score and updates it based on game events (e.g., eating pellets, eating ghosts). Displays the score to the player on the game screen.

### 1.1.10 Sound Effects:

Plays sound effects to enhance the gaming experience (e.g., eating pellets, colliding with ghosts). Uses a sound library (e.g., Java Sound API) to play the sound effects.

## Frameworks and Software

### 2.1 Frameworks

#### 2.1.1 Java Development Kit (JDK)

Required for java development. Includes the Java Runtime Environment (JRE) aswell as tools that are used for debugging and monitoring java applications.

#### 2.1.2 Graphics Libraries

Uses Java2D and Swing components to create graphics for more advanced rendering.

#### 2.1.3 Sound Libraries

Utilizes the Java Sound API with .wav format sound files in order to play sound effects and music withing the game.

### 2.2 Software

#### 2.2.1 Eclipse IDE

An integrated development environment (IDE) for Java development. Eclipse provides tools for writing, compiling, debugging, and running Java programs.

#### 2.2.2 JUnit

A unit testing framework for Java. Junit is used to write and run tests for your game code to ensure its correctness.

#### 2.2.3 Github

A version control system for tracking changes in your codebase. Git can help you manage and collaborate on your Pac-Man game project with ease.