SIMON FRASER UNIVERSITY
Department of Computing Science
CMPT 276: Assignments

Project, Phase 1: Requirement Engineering and Design

Deliverables

You will be working on your group repository on SFU's GitHub. Each team member should contribute continually to all phases of the project.

Create a "Design" directory in your team's repository, for all the deliverables of this phase. We will mark the history and outcome of the assignment on SFU's GitHub on the specified deadline. See Canvas for deadlines.

Project Overview

The goal of this project is to design, implement, test, and evaluate an arcade-style 2D game. The player controls the main character from the start point to the end point through barriers and the enemies on the board. The main character should collect the rewards that are placed at various places on the map. The score of the player, which is the accumulation of all rewards, and the playtime are shown on the screen. The player wins by reaching the "exit" cell after collecting all "regular" rewards on the board. The player loses if the main character encounters or is caught by a moving enemy, or the overall score becomes negative. Each moving character can move at most one cell at each "tick" of the game, as described below.

The main entities of the game are:

- **Main Character.**
  - The game has **one** main character, which is controlled by the player through the keyboard.
  - The initial location of the main character is the start cell on the board map.
  - The main character can move up/down/left/right to an adjacent cell, if there are no barriers on the target cell. The player can move at most one cell at each "tick" of the game. The main character will not move if the player doesn't press a valid key, or if the move is not valid (i.e., a move to a cell that is blocked by a wall or a barrier). Even in such cases, the enemies will move, as explained in the "Enemies" section below.
  - If the main character moves to a cell that contains a reward, it collects the amount of reward, and the reward is removed from the field.
  - If the main character moves to a cell that contains a punishment (unanimated enemy), it gets penalized by the amount of the punishment, and the punishment is removed from the board. If the total score of the player drops below 0 as a result, the player loses.

  - If the main character moves to a cell that contains a moving enemy, or an enemy catches up to the main character, the game is over and the player loses.
  - There can be at most one moving character (main character or moving enemy) on each cell at a time.

- **Enemies.** The game should have a minimum of two types of enemies, with multiple instances of each type. First type of enemies should be animated, moving at each "tick" of the game. Another type of enemies is a penalty or a punishment, placed on a cell on the board.
  - **Moving Enemy.**
    - The moving enemies move one cell up/down/left/right at each tick. They move towards a direction that makes them closest to the current position of the main character.
    - If moving enemies pass through cells that contain rewards or punishments, if won't affect the awards/punishments.
    - The enemies can't go through the walls/barriers either
  - **Punishment.** It the main character moves to a cell that contains a punishment, it will be penalized by the amount of the punishment. The player loses if the punishment causes the overall score to become negative.
- Rewards. The game should included a minimum of two types of rewards.
  - **Regular Reward.** When the game starts, the board is populated with multiple "regular" rewards. The player has to collect all regular rewards to win game. To collect a reward, the player should move the main character to the cell containing the reward, which lead to the reward being claimed (removed from the map) and the reward amount being added to the total score of the player, displayed on the screen.
  - **Bonus Reward.** The bonus rewards are not necessary for finishing the game, but contain a higher reward compared to the regular rewards and collecting them improves the final score of the player. A bonus reward appears randomly during the game, and disappears after a while (few ticks).
- **Barriers.**
  - The board is surrounded by walls from four sides. There are only two openings for the start and end points. There are also other walls on the board (series of blocked cells), creating a maze-like map.
  - There are other individual barriers on the map cells, which also block the movement of the main character and enemies to those cells.
- **Board.** The game board is a 2D grid, surrounded by walls.
  - An initial map for the board should be loaded when the player starts the game, showing the placement of the walls/barriers, "regular" rewards, inanimate punishments, as well as the initial locations of the main character and the moving enemies.

- The board has a start point, which is a cell on the boundary wall of the board. The main characters starts the game there. It also has an end cell, which is another cell on another boundary wall of the board. The player can win the game by guiding the main character to the end cell after collecting all regular rewards.
- – The game screen shows the current score and the time that has passed since the start of the game. The final score and time are shown for a winning player.

By the end of the project, you will complete this game as a Desktop application written entirely in Java.

Phase 1: Requirements and Design

During the first phase of the project, you will focus on requirements engineering and designing the game. You will create a set of use cases that describe the behaviour of the system from a player's perspective.

Using the use cases, you will then create a design that reflects the architecture of your system. This design should include: (i) mockups of your system's user-interface, and (ii) UML class diagrams for the main components of your system and the relationships between them, and for key data structures that are manipulated by and/or exchanged between your components.

Deliverables for Phase 1

In this phase, you are required to produce the following:

1. A one-page statement describing your overall plan and the description of your customized game,
2. Use cases that describe different aspects of the system's functionality (following the style used in the Pressman & Maxim textbook),
3. One or more mockups of your user-interface (maximum of 2 pages PDF), and
4. UML class diagrams reflecting main components and data structures (classes, packages, relations, etc.).

All documents should be submitted through your SFU GitHub repository in PDF format. Create a "Design" directory in your team's repository. All deliverables for this phase should be completed by the deadline under this directory. All team members need to collaborate and jointly contribute to this phase (and all future phases).