Use Cases - Group #13

Xuyun Ding, Denzel Tjokroardi, Kyle Tsia, Yihao Wang

Case 1:

Use case: Initialize Game.
Primary actor: Computer User.
Goal in context: The user chooses what he wants to do with the game.
Level: User goal.
Preconditions: The user wants to play the game.
Trigger: The user decides to load the game from the computer.
Main Scenario:

1. The game application has started, and the main menu is displayed.
2. The user chooses one of the main menu options: Start, and How To Play.
3. Program continues onto the use case of whatever the user chose.

Extensions:

1. The user chooses an invalid option.
2. The program loops until the user chooses a valid option from the main menu.

Priority: Essential, must be implemented.
When available: Right after the game app is opened.
Frequency of use: Whenever the player chooses to return to the main menu or the program
Channel to actor: Via keyboard and computer screen (GUI).
Open issues:

1. Should there be more scenario (game options from the main menu, such as play a saved game, view high score etc.)?
2. Should there be more situations/cases in the **Extension** part?

Case 2:

Use case: Gameplay (Player Movement).
Primary actor: The Player (User).
Goal in context: The player wants to move the Hero icon around the board in order to collect rewards and avoid enemies.
Level: System goal.
Preconditions: The player has chosen to start the game and the game initialized.
Trigger: The player pushes the "Up" button to move the hero up, the "Down" button to move the hero down, the "Left" button to move the hero left, or the "Right" button to move the hero right.
Main Scenario:

1. The player pushes one of the movement buttons.
2. The hero icon moves in the direction that the player entered.
3. The hero icon continues to move in the same direction until the player tries to move the hero icon into a cell that has a wall or barrier.

Extensions:

1. An exception is thrown while getting the player input. Catch exception and continue listening to future player inputs.
2. A moving enemy icon (failing course) moves into the cell where the hero icon is before the player inputs a movement command.
   a) A message is displayed showing that the current game is over, and the player loses.
3. The player tries to move the hero into a cell that has a regular reward icon (mandatory assignments) in it.
   a) The hero icon successfully moves into the cell.
   b) The regular reward icon is removed from the GUI.
   c) A corresponding score is added to the total score and updated on the screen.
4. The player tries to move the hero icon into a cell that has a bonus reward icon (optional achievements) in it.
   a) The hero icon successfully moves into the cell.
   b) The bonus reward icon is removed from the GUI.
   c) A higher corresponding score is added to the total score and updated on the screen
5. The player tries to move the hero icon into a cell that has a moving enemy (failing course) icon in it.
   a) A message is displayed showing that the current game is over, and the player loses.
6. The player tries to move the hero icon into a cell that has a stationary enemy (punishment/distractions) icon in it.
   a) The hero icon successfully moves into the cell.
   b) The stationary enemy icon is removed from the GUI.
   c) A corresponding score is deducted from the total score and updated on the screen.
      1) If the total score becomes negative, then a message is displayed showing that the current game is over, and the player loses.
      2) Otherwise, waiting for the next player movement command.
7. The player tries to move the hero icon into the exit cell.

    a) The hero icon successfully moves into the cell.
        1) Exit cell only spawns after complete collection of regular rewards, thus the game proceeds to the next stage.

Priority: Essential, must be implemented.
When available: Right after the start selected.
Frequency of use: Once per game round.
Channel to actor: Via keyboard and computer screen (GUI).
Open issues:

1. Should there be more instances/situations throughout the whole playthrough of one game round?
2. Should there be more detailed messages to be displayed on screen in each situation?

Case 3:

Use case: Check Game Instruction.
Primary actor: Computer User (player).
Goal in context: The player learns the game instructions.
Level: User goal.
Preconditions: The game application has been launched and the player is at the main menu.
Trigger: The player wants to know how to correctly play the game.
Main Scenario:

1. The player selects "How To Play" from the main menu.
2. A screen shows up with several messages describing how the game works (hero, enemies, win/lose conditions, game board, movement controls…).
3. The player hit the "Return" button.
4. The player is taken back to the main menu.

Extension:

1. If there are submenus for each category of the game instruction.
    a. The player hits each submenu button to view the corresponding instruction message.
    b. Within each submenu screen, there will be a button for the player to return to the previous menu.

Priority: Optional. Only if we decide to add such feature into the game (not necessary, but better to have).
When available: Once the game application has been opened.
Frequency of use: At least once per game application launch.
Channel to actor: Via keyboard/mouse and computer screen (GUI).
Open issues:

1. Should icons/legends be added to the instruction messages to better illustrate the game mechanics?
2. Should those instruction messages be divided and included into submenus? If so, the **Scenario** may need some changes.

Case 4:

Use case: Pause Game.
Primary actor: Computer User (player).
Goal in context: The player is able to pause the game and later resume the game when hitting the resume button
Level: User goal.
Preconditions: The game must be currently running and being played.
Trigger: The player presses the pause button.
Main Scenario:

1. The player hits the space button during the game.
2. A menu is displayed asking if the player wants to resume the game.
3. The program waits for the player input.
4. The player chooses to resume the game.
5. The game is restored to the state it was in prior to the pause.

Extensions:

1. The player is given the option to resume or exit the game.
    a) If the player chooses to resume the game, then jump to scenario 5.
    b) Otherwise, jump to Exit Game use case.

Priority: Optional, only if we decide to add the Pause Game functionality into the game.
When available: When a game round has started.
Frequency of use: Several times per game round (whenever the players decides to hit the pause button during a game).
Channel to actor: Via keyboard and computer screen (GUI).
Open issues:

1. Should there be more options or information on the pop-up menu after the player pause the game?
2. What could be a good choice for the pause and resume button, and should the prompt be displayed on the screen?

Case 5:

Use case: Change Controls.
Primary actor: Computer User (player).
Goal in context: The player decides to change controls to more desirable set.
Level: User goal.
Preconditions: The game application has been started and at the main menu.
Trigger: The player selects Change Controls from the main menu.
Main Scenario:

1. The player selects Change Controls from the main menu.
2. The program gives the prompt for changing the controls with the options of "WASD keys" and "Four arrow keys".
3. The player is given a Success Message and is returned to the main menu.

Extension:

1. The player does not respond.
    a) System will have a timeout (to be decided) and return the player to the main menu with the default controls restored.
2. Error on saving controls.
    a) System will display a message that an Application error has occurred.
    b) The player will be taken back to the main menu and the default controls are restored.

Priority: Optional, only if we decide to add the Change Controls functionality into the game (not necessary, but better to have).
When available: At the start of one game round from the main menu.
Frequency of use: Once per game round.
Channel to actor: Via keyboard/mouse and computer screen (GUI).
Open issues:

1. Should we give the player fully customizable game controls instead of predefined ones?
2. If so, what could the scenario and extension be?

<u>Case 6:</u>
Use case: Exit Game.
Primary actor: Computer User (player).
Goal in context: Based on the type of exit made by the player, the program should be able to know which step to follow.
Level: User goal.
Preconditions: The player has hit the exit button, lost the game or beat the game, and the game finishes accordingly.
Trigger: The player presses the button, loses the game or wins the game.
Main Scenario:

1. The player has exited the game.
2. The device has returned to the state it was in before the game was played.

Extension:

1. The hero icon has been caught by a moving enemy (failing course).
2. The total score has become negative.
3. The player completes and wins the game.
4. Completion screen is displayed.

Priority: Essential, must be implemented.
When available: Whenever the player decides to exit the game, loses the game or beats the game.
Frequency of use: At least once per game round.
Channel to actor: Via keyboard/mouse and computer screen (GUI).
Open issues:

1. Should the game exit automatically after the player loses/wins the game?
2. If not, what could the prompt be?
3. Should there be an option for the player to exit the game to the main menu or just directly to the desktop?