Comprehensive Exercise Report

Team 31 of Section ADB

Muslim Berat CANPOLAT 211ADB068
Emre OZSOY 230ADB073
Abdulkadir ARSLAN 230ADB074
Baturalp BURMAOGLU 211ADB042

Requirements/Analysis

Week 2

Journal

The following prompts are meant to aid your thought process as you complete the requirements/analysis portion of this exercise. Please respond to each of the prompts below and feel free to add additional notes.

- After reading the client's brief (possibly incomplete description), write one sentence that describes the project (expected software) and list the already known requirements.

  Project Description:
  The software is a digital version of the game "4 Connect," where players take turns dropping colored coins into a grid with the goal of connecting four of their own coins vertically, horizontally, or diagonally before their opponent does.

  Known Requirements:
  - Implement a grid/board for gameplay.
  - Allow two players to take turns dropping coins of different colors into the grid.
  - Detect when a player has connected four of their own coins in a row and announce the winner.
  - Provide a user interface for players to interact with the game.

- After reading the client's brief (possibly incomplete description), what questions do you have for the client? Are there any pieces that are unclear? After you have a list of questions, raise your hand and ask the client (your instructor) the questions; make sure to document his/her answers.

  Questions for the Client:

  - Are there any specific rules or variations of the game that need to be implemented?
  - Should the game have options for different difficulty levels?
  - Are there any specific visual or design preferences for the user interface?

- Does the project cover topics you are unfamiliar with? If so, look up the topics and list your references.

  Topics We're Unfamiliar With and References:

  Some of our team members have independent expertise in creating small game projects, which is beneficial to our team. We're not specialists, but we know the fundamentals of game development. But there's a chance we'll face difficulties beyond our experience. We'll leverage Stack Overflow's Game Development section, Unity Learn, Unreal Engine Documentation, GameDev.net, and Game Developer (website) as online learning and guidance tools to handle this.

- Describe the users of this software (e.g., small child, high school teacher who is taking attendance).

  Users of the Software:
  - Players of different ages enjoy strategic board games.
  - It could be anyone from children to adults who have an interest in competitive or casual gaming.

- Describe how each user would interact with the software.

  User Interaction:
  - Each player would take turns selecting a column in the grid to drop their colored coins.
  - They would observe the movement of coins on the grid and strategize to connect four of their own coins while blocking their opponent from winning the game.

- What features must the software have? What should the users be able to do?

  Required Features:
  - Implement a grid game board.
  - Allow players to drop coins into columns and detect when a player has connected four coins in a row.
  - Provide a user-friendly interface for gameplay, including options for starting a new game, quitting, and possibly adjusting settings.

- Other notes:

  Features such as an undo button for players to correct their moves, a visual indication of whose turn it is, and sound effects or animations to enhance the gaming experience. Additionally, accessibility features like colorblind mode may be worth exploring.

Software Requirements

The project aims to develop a digital version of the game "4 Connect," where two players compete to connect four of their own coins in a row either vertically, horizontally, or diagonally within a grid. The software will provide a UI for players to interact with the game, including options for starting new games, quitting, and possibly adjusting settings. The game will have various players of ages and skills, offering an engaging experience for those interested in board games which offer a lot of fun.

Requirements:
1. Grid/Game Board:
   - The software must implement a grid or game board with dimensions suitable for gameplay.
   - The grid must be visually represented on the UI.

2. Player Interaction:
   - Players must be able to take turns dropping coins of their own into columns of the grid.
   - The software must accurately display the movement of coins on the grid.

3. Winning Condition:
   - The software should detect when a player has successfully connected four of their own coins in a row and declare the winner.

4. User Interface:
   - The UI should be easy to navigate.
   - Options for starting a new game and quitting should be accessible to the players.
   - Settings options, if implemented, should allow players to adjust game parameters such as audio level.

5. Additional Features:
   - Visual reminders, such as highlighting the current player's turn, should be incorporated into the interface.
   - Sound effects may be included to enhance the gaming experience.
   - Consideration should be given to accessibility features, such as a colorblind mode, to ensure inclusivity for all players.

User Stories:
- Hi, I am Andrejs, I want to be able to drop my colored coins into the grid and see it appear in the designated column without having bugs such as coins jumping after hitting the designed place for it.
- Hi, my name is Jane, I want the game to accurately detect when I've connected four of my own coins in a row and declare me as a winner without having any bugs and errors.
- Hey, I am Juris, I want to have options for starting a new game, quitting, and adjusting settings to tailor the game to my preferences with a good UI design.

These requirements and user stories will guide the development process, ensuring that the software meets the expectations of both the client and the end users.

Black-Box Testing

Instructions: Week 4

Journal

Remember: Black box tests should only be based on your requirements and should work independent of design.

The following prompts are meant to aid your thought process as you complete the black box testing portion of this exercise. Please review your list of requirements and respond to each of the prompts below. Feel free to add additional notes.

- What does input for the software look like (e.g., what type of data, how many pieces of data)?
    - The primary inputs include player actions, which consist of selecting columns in the grid to drop their colored coins. At any given turn, the input is singular the selection of one column by a player to drop a coin into. But, the game will process numerous such inputs throughout its course, one for each turn taken by the players.
- What does output for the software look like (e.g., what type of data, how many pieces of data)?
    - The outputs include the state of the game board after a coin is dropped audio feedback for player actions and notifications such as highlighting the current player's turn the winner announcement or a draw notification if the board fills up without a winner. output is multi-faceted after each input it includes the new state of the grid and possibly a game state update.
- What equivalence classes can the input be broken into?
    - Valid Inputs: Any column selection that is not full and within the grid's bounds.
    - Invalid Inputs: Selecting a column that is already full or outside the grid's bounds.
    - Special Inputs: The last possible move that could either result in a win, a draw, or fill the last remaining space without changing the game state.
- What boundary values exist for the input?
    - Lower Boundary: Selecting the first column when it's not full.
    - Upper Boundary: Selecting the last column when it's not full.
    - Full Column: Attempting to place a coin in a column that is already full should not be allowed and should prompt the player to select again.
- Are there other cases that must be tested to test all requirements?
    - Winning Condition Checks: Directly horizontal vertical and diagonal connections of four coins need thorough testing.
    - Draw Condition: The board is filled without any player winning.
    - UI Navigation: Transitioning between game states and accessing settings or quitting the game.
    - Accessibility Features: Testing colorblind mode to ensure it's effectively implemented.
- Other notes:
    - Performance Testing: Ensuring the game runs smoothly without lag during coin drop animations and state transitions.
    - Usability Testing: Feedback from players of various age groups to ensure the game is engaging and the UI is intuitive.
    - Compatibility Testing: Testing on different devices and screen sizes to ensure consistent gameplay experience.

Black-box Test Cases

Use your notes from above to complete the black-box test plan section of the formal documentation by writing black box test cases (other than actual results since no program currently exists). Remember to test each equivalence class, boundary value, and requirement.

Test ID	Description	Expected Results	Actual Results
TC-01	Drop coin in an empty column.	Coin appears at the bottom position of the selected column.	
TC-02	Drop coin in a column that is partially filled.	Coin on top of the last dropped coin in the selected column without any overlap.	
TC-03	Attempt to drop coin in a column that is fully filled.	System prevents action and prompts user to select another column.	
TC-04	Connect four coins horizontally.	Game detects the connection, announces the player as the winner and ends the game.	
TC-05	Connect four coins vertically.	Game detects the connection announces the player as the winner and ends the game.	
TC-06	Connect four coins diagonally.	Game detects the connection, announces the player as the winner and ends the game.	
TC-07	Fill the board without any player connecting four coins.	Game declares a draw and offers an option to start a new game.	
TC-08	Navigate to the game settings from the main menu.	The settings menu is displayed allowing the user to adjust game parameters such as audio level.	
TC-09	Start a new game after finishing a game.	A new game starts with an empty grid and reset scores.	
TC-10	Use the quit option to exit the game.	The game closes and exits to the system or main menu, depending on platform capabilities.	
TC-11	Drop a coin at the lower boundary (first column) when it's not full.	Coin appears at the bottom position of the first column.	
TC-12	Drop a coin at the upper boundary (last column) when it's not full.	Coin appears at the bottom position of the last column.	
TC-13	Testing colorblind mode (if implemented) for visibility.	Game visuals adjust to colorblind mode ensuring all players can distinguish between different colored coins.	
TC-14	Observe visual and audio feedback during a player's turn.	Visual indicators highlight the current player's turn and sound effects are played during coin drop.	
TC-15	Attempt to perform an action outside the boundaries of the game grid.	The game ignores the action or displays an error message, prompting for a valid move within the grid boundaries.	

Design

Instructions: Week 6

Journal

Remember: You still will not be writing code at this point in the process.

The following prompts are meant to aid your thought process as you complete the design portion of this exercise. Please respond to each of the prompts below and feel free to add additional notes.

- List the nouns from your requirements/analysis documentation.
    - <<Insert answer>>
- Which nouns potentially may represent a class in your design?
    - <<Insert answer>>
- Which nouns potentially may represent attributes/fields in your design? Also list the class each attribute/field would be a part of.
    - <<Insert answer>>
- Now that you have a list of possible classes, consider different design options (***lists of classes and attributes***) along with the pros and cons of each. We often do not come up with the best design on our first attempt. Also consider whether any needed classes are missing. These two design options should not be GUI vs. non-GUI; instead you need to include the classes and attributes for each design. Reminder: Each design must include at least two classes that define object types.
    - <<List at least two design options with pros and cons of each>>
- Which design do you plan to use? Explain why you have chosen this design.
- List the verbs from your requirements/analysis documentation.
    - <<Insert answer>>
- Which verbs potentially may represent a method in your design? Also list the class each method would be part of.
    - <<Insert answer>>
- Other notes:
    - <<Insert notes>>

Software Design

<<Use your notes from above to complete this section of the formal documentation by planning the classes, methods, and fields that will used in the software. Your design should include UML class diagrams along with method headers. ***Prior to starting the formal documentation, you should show your answers to the above prompts to your instructor.***>>

Implementation

Instructions: Week 8

Journal

The following prompts are meant to aid your thought process as you complete the implementation portion of this exercise. Please respond to each of the prompt below and feel free to add additional notes.

- What programming concepts from the course will you need to implement your design? Briefly explain how each will be used during implementation.
    - <<Insert answer>>
- Other notes:
    - <<Insert notes>>

Implementation Details

<<Use your notes from above to write code and complete this section of the formal documentation with a README for the user that explains how he/she will interact with the system.>>

Testing

Instructions: Week 10

Journal

The following prompts are meant to aid your thought process as you complete the testing portion of this exercise. Please respond to each of the prompts below and feel free to add additional notes.

- Have you changed any requirements since you completed the black box test plan? If so, list changes below and update your black-box test plan appropriately.
    - <<Insert answer>>
- List the classes of your implementation. For each class, list equivalence classes, boundary values, and paths through code that you should test.
    - <<Insert class>>
        - <<Insert needed tests>>
    - <<Insert class and tests for each class>>
- Other notes:
    - <<Insert notes>>

Testing Details

<<Use your notes from above to write your test programs and complete this section of the formal documentation by creating a list of your test programs along with descriptions of what they are testing. You will also complete the black-box test plan by running the program and filling in the Actual Results column.>>

Presentation

Instructions:Week 12

Preparation

The following prompts are meant to aid your thought process as you complete the presentation portion of this exercise. It is recommended that you examine the previous sections of the journal and your reflections as you work on the presentation as it is likely that you have already answered some of the following prompts elsewhere. Please respond to each of the prompts below and feel free to add additional notes.

- Give a brief description of your final project
    - <<Insert answer>>
- Describe your requirement assumptions/additions.
    - <<Insert answer>>
- Describe your design options and decision. How did you weigh the pros and cons of the different designs to make your decision?
    - <<Insert answer>>
- How did the extension affect your design?
    - <<Insert answer>>
- Describe your tests (e.g., what you tested, equivalence classes).
    - <<Insert answer>>
- What lessons did you learn from the comprehensive exercise (i.e., programming concepts, software process)?
    - <<Insert answer>>
- What functionalities are you going to demo?
    - <<Insert answer>>
- Who is going to speak about each portion of your presentation? (Recall: Each group will have ten minutes to present their work; minimum length of group presentation is seven minutes. Each student must present for at least two minutes of the presentation.)
    - <<Insert answer>>
- Other notes:
    - <<Insert notes>>

<<Use your notes from above to complete create your slides and plan your presentation and demo.>>