Comprehensive Exercise Report

Team Team-21 of Section ADB

Ibrahim Velizade 211ADB065

Farid Aghazada 211ADB092

[**Requirements/Analysis**](#_uwgqwd5ezv2w) **2**

[Journal](#_lsityg2iq9m6) 2

[Software Requirements](#_2h0vru1u2mla) 3

[**Black-Box Testing**](#_prhaxdxmf8n8) **4**

[Journal](#_18f11w613jft) 4

[Black-box Test Cases](#_2xn4jzot820y) 5

[**Design**](#_24fdizefyocn) **6**

[Journal](#_esp2ocs9j6bk) 6

[Software Design](#_aifbl1x6rddt) 7

[**Implementation**](#_hya8f3jqkba6) **8**

[Journal](#_acupzfhai7gz) 8

[Implementation Details](#_ojhtwkms2z3b) 9

[**Testing**](#_3qvya3vi836q) **10**

[Journal](#_ckfs4xbl5pyr) 10

[Testing Details](#_bzt1547yxzxi) 11

[**Presentation**](#_hdjvrbf45b1p) **12**

[Preparation](#_xbiquwtmf36n) 12

[**Grading Rubric**](#_u0hfnmdgusmf) **13**

# 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.
  + Standard Connect 4 game with GUI that is played with 2 people locally and follow the traditional Connect 4 rules
    - Graphical User Interface
    - Game rules
    - Played locally
    - Ability to start a new game or exit the application
    - 2-player
* 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.
  + Q: Are there any specific design preferences for the graphical user interface (GUI)? A: It should resemble the physical game set, i.e., the color of the disks
  + Q: Are there any specific design preferences for the graphical user interface (GUI)? A: No, you can choose what you wish
  + Q: On which devices should a user be able to play this game? A: It should be a desktop application
  + Q: Are there any additional features or functionalities you would like to include other than the basic requirements listed? A: It is optional
* Does the project cover topics you are unfamiliar with? If so, look up the topics and list your references.
  + GUI implementation
  + Desktop game development
  + Game algorithms
* Describe the users of this software (e.g., small child, high school teacher who is taking attendance).
  + Everyone who is familiar with game rules
* Describe how each user would interact with the software
  + Players will be able interact with the game by using mouse/touchpad to click relevant buttons in the game
* What features must the software have? What should the users be able to do?
  + To restart the game
  + To see the game outcomes (win/lose)
  + Users should be able to make moves by dropping tokens into columns.
* Other notes:
  + Overall it should give the impression of the playing with the physical game set

## Software Requirements

The software is the web application of the classic Connect 4 game. Connect 4 is a two-player strategy game where the goal is to be the first to form a line of four of your colored discs in a row, horizontally, vertically, or diagonally on a grid. The game is played on a vertical grid with 6 rows and 7 columns, totaling 42 spaces. It is played with red and yellow disks. Players take turns dropping one of their colored discs from the top into any of the columns. The disc then falls to the lowest available space in that column. To win the game players need to build their own line of four while also strategically blocking their opponent from forming their own line. The game ends when one player successfully connects four of their discs in a row, and that player is declared the winner. If the board fills up without a player connecting four discs, the game is a draw.

In the main page where the gaming takes place, there is a board and disks inside the spaces like in the physical game. During the game the players take turns clicking on the column where they want to drop the disk (red or yellow depending on whose turn it is) and it continues till the end of the game. When the game ends users have a choice to exit the game or replay.

The software requirements are the following:

* Game Interface: Develop a user-friendly interface that allows players to interact with the game, i.e., drop discs into columns, and view the game state in real-time.
* Game Logic: Implement the rules of Connect Four
* Responsive Design: Ensure the web application is responsive and works across various devices and screen sizes
* Game State Management: Implement a system to manage and update the game state, including storing the position of discs on the grid and handling player turns.
* User Experience Enhancements: Provide visual feedback when hovering over columns to indicate where the disc will drop. Animations (such as disc dropping and winning sequences) can enhance the user experience.
* Additional Features: Ability to restart game at any given moment.

# 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)?
  + Mouse click events: player actions, such as clicking on a column to drop a disc or to restart the game. The number of mouse clicks depends on the progress and the state of the game.
* What does output for the software look like (e.g., what type of data, how many pieces of data)?
  + Visual feedback on the game interface: the updated game grid after each player's turn, indicating whose turn it is, and displaying messages for game outcomes (win, draw).
* What equivalence classes can the input be broken into?
  + Valid input: actions that follow the game rules, such as clicking on a valid column to drop a disc.
  + Invalid input: actions that violate the game rules, such as attempting to drop a disc in a column that is already full or clicking outside of game area.
* What boundary values exist for the input?
* Position of the mouse click: it should be on the game grid that consists of 6 rows and 7 columns or on the other available buttons
* Are there other cases that must be tested to test all requirements?
  + Testing if restarting the game works.
  + Testing if the user experience enhancements (e.g., hovering on columns to indicate where the disc will drop) work properly
  + Testing the responsiveness of the web application across various devices and screen sizes to ensure consistent gameplay
* Other notes:
  + Tests can be done in different browsers to see if there is any difference in the results

## 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** |
| 1 | Valid input: Clicking on a valid column to drop a disc. | Disc is dropped in the selected column. The Game grid is updated accordingly. |  |
| 2 | Invalid input: Clicking on a column that is already full. | No disc is dropped. Player receives a notification indicating the invalid move. |  |
| 3 | Invalid input: Clicking outside of the game area and buttons | No action is taken. The game state remains unchanged. |  |
| 3 | Testing the response when the grid is full | Game ends in a draw. Players are notified of the draw outcome. |  |
| 4 | Testing if restarting the game works. | Game state is reset to the initial state. Players can start a new game. |  |
| 5 | Testing user experience enhancements: hovering over columns to indicate where the disc will drop. | Visual feedback is provided when hovering over columns, indicating where the disc will drop. |  |
| 6 | Testing responsiveness across various devices and screen sizes | Game interface adjusts smoothly to different screen sizes. Gameplay remains consistent and enjoyable across devices. |  |

# 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.>>