Comprehensive Exercise Report

Team ALBcreators

<<Group Member names and unity IDs

Simeon Flamuraj-230ADB027

Adela Muskollari-230AIB019>>

NOTE: You will replace all placeholders that are given in <<>>

[**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.
  + <<The client described a development of a game called “connect four” which will require 2 players and two different colored stones to play on a grid board.>>
    - <<At the game the first one who connect four stones in any way in same colors, for example one player red and another yellow, will win the match and gain some points and who ever reaches 100 points will win the game, 1 match is 20 points in case of a tie 10 points for each 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.
  + <<1-Is there any way to get bonus points?

2-Is there a specific number of stones for the players?

3-Is there any chance to lose points?>>

* Does the project cover topics you are unfamiliar with? If so, look up the topics and list your references.
  + <<Fairly familiar-We have played a similar game before and looked it in the references below:
  + 1-https://www.wikihow.com/Play-Connect-4
  + 2-https://instructions.hasbro.com/en-my/instruction/connect-4-game>>
* Describe the users of this software (e.g., small child, high school teacher who is taking attendance).
  + <<Aimed to 6+ audience >>
* Describe how each user would interact with the software
  + <<The user will interact through a web page>>
* What features must the software have? What should the users be able to do?
  + <<First of all, it would be interactive and the player is able to choose where he will put the stones to make his own strategy to win.>>
* Other notes:
  + <<Insert notes>>

## Software Requirements

<<

* Software Requirements:

1. In the game would be 56 stones 28 red ones and 28 yellow ones.
2. The game must have a board with size 7x8.
3. The game mode is only two players.
4. Before every turn the code is checked for a win or draw condition.
5. After the win or draw the points are registered to the right player,(win-20 points, draw-10 points each)
6. The game constantly check if a player has reached maximum points therefor the winner.(100 points in case of a tie which is rare an extra round will start to determine the winner)

* User stories:

1. Like a player, I want to choose where I put my stones.
2. As a player, I want to see how many points I have.
3. As a player, I want to know the maximum points to win.
4. As a player, I want to start a new game after reaching maximum points.
5. As a player, I want to exit the game mid-way.

>>

# 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 input would be from the players to choose where they put the stones in the board which is the size 7X8, but the stones will be always at the last free grid from the bottom, till they connect 4 stones with the same color.>>
* What does output for the software look like (e.g., what type of data, how many pieces of data)?
  + << After every turn the new stone will be added to the board represented as matrix in the code meanwhile checking for the win or draw>>
* What equivalence classes can the input be broken into?
  + <<Board:7collumns X 8 rows
  + Stones:28 stones each player(red and black)
  + Win\_condictions: horizontal, vertical, diagonal
  + Draw\_conditions
  + Menu
  + Number of players:2
  + >>
* What boundary values exist for the input?
  + << fixed board size
  + Number of the players only 2
  + No timer >>
* Are there other cases that must be tested to test all requirements?
  + <<Testing for not correct coordinates
  + Testing for invalid input
  + >>
* Other notes:
  + <<Insert notes>>

## 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 | Testing the displayBoard | The boar 7X8 will be printed | N/A |
| 2 | Testing the Graphical interface(GUI) | The menu will pop up for starting the game | N/A |
| 3 | Starting a new game | A new game will start and player 1 would be able to put the first move | N/A |
| 4 | Testing when the turns are switched | When the player 1 will finish his turn the player 2 will be able to put his stone | N/A |
| 5 | Testing the win for each player | When the player 1 or the player 2 will connect 4 he will win points and the next round would start | N/A |
| 6 | Testing the draw | When the players will ran out of stones or moves the fame will end with a tie and the next round will start | N/A |
| 7 | Testing the points for the win conditions for each player | When the player wins the score of that round will be shown and registered | N/A |
| 8 | Testing the points for the draws condition | When the game ends with draw a same amount of the points is added to both players | N/A |
| 9 | Testing when player 1 or player 2 have similar points at maximum points an extra round will start | When the players at the maximum stores ends up with similar points the extra round will decide who is the winner | N/A |
| 10 | Testing when player 1 or player 2 reach the maximum points | When the player reach the maximum points he will win the game and the menu will pop up for starting a new game | N/A |

# 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.
  + User
  + Us
  + Player
  + Product
  + Configuration
  + System
  + Game
  + Stone
  + Account
* Which nouns potentially may represent a class in your design?
  + User
  + Game
* Which nouns potentially may represent attributes/fields in your design? Also list the class each attribute/field would be a part of.
  + Name
  + Points
  + Settings
  + View points
* 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>>
  + Player
  + Game

Pros:

* Simple and straightforward with only two classes
* Matches a real game that is created by the game itself and the player
* Can be easy to implement for a simple game

Cons:

* May not handle more complex game algorithms
* May not include all classes and attributes to make the game more satisfactory for players
* Which design do you plan to use? Explain why you have chosen this design.  
  The system is relatively small and simple, and a more complex design would be overkill. The system is a game so these two classes are the most relevant ones. Most players would like a simple game that it can also be played by different group ages.
* List the verbs from your requirements/analysis documentation.
  + For the noun Us the verb is designing (since we are designing the system)
  + For the noun User the verbs are playing, views the points of the game
  + For the noun Game verb is storing the player’s game information
* Which verbs potentially may represent a method in your design? Also list the class each method would be part of.
  + startGame(): This method is part of Game class and would allow a new game to start
  + finishGame(): This method is also part of Game class and would allow to finish the game
  + lookPoints(): This method is part of Players class and would allow them to look at the points collected
* 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.****>>*

mentioned for the design there will be two classes Player and Game but also additional classes might be implemented later. This design will be simple and straightforward, every age group will be able to play it.  
The moment the play enters the game they can start playing by pressing play, if they wish to end the game sooner, they can press finish game. There will also be the option that they can view the points they have received for each game they have won.

|  |
| --- |
| User |
| * Name * User Id |

|  |
| --- |
| Player |
| * lookPoints() * playsGame() |

|  |
| --- |
| Game |
| * startGame() * finishGame() |

# 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.>>