

## CPSC 224 - Final Project - Part 2

### Project Planning and Yahtzee Design

Group Members: Jackie Ramsey, Jesse Adams, John Stirrat, Katie Imhof

## Part 1: Project Description

For this project, we are building a game called Yahtzee. The version that we are building is the original, multiplayer game where players are competing to get the highest total score and win. The way the game works is by each player taking turns rolling dice and adding up the total for specified score combinations. These score categories can be found on the provided score card that contains different types of dice combinations with an associated score. On each turn, the player will roll dice up to 3 times in order to get the highest scoring combination for one of the 13 categories on the scorecard. There are a total of 5 dice for each turn. On the first roll, the player must roll all the dice. However, for the second and third roll, the player can decide which dice they would like to keep or reroll by either inputting “y” or “n” for each die. The purpose of this is so that the player can strategize which score category they are going to try and get the most points for.

Once the player finishes rolling, they must place the score or zero in one of the available categories by inputting the keycode associated with the score category. Every turn, it is required that a score category must be filled. Additionally, each category can only be used one, so players must strategize where to put their points in an attempt to have the best score. In addition to the 13 categories on the scorecard, there is a bonus that will be added to the final total of the scorecard if the requirements are met. One significant score category to be mindful of when playing this game is the Yahtzee category. If a player gets 5 of the same number, the Yahtzee category can be filled with 50 points which is a huge advantage.

When the player is finished with their turn, the next player plays their turn and so on. This process is repeated for each player until all players have filled the 13 categories in their score card. Keeping track of progress is important, so at any time, players can view the current scorecards by inputting the associated command. Once the score cards of all the players are filled, the game is over. The total score is calculated for each player's score card, and the player with the highest total score will be determined as the winner.

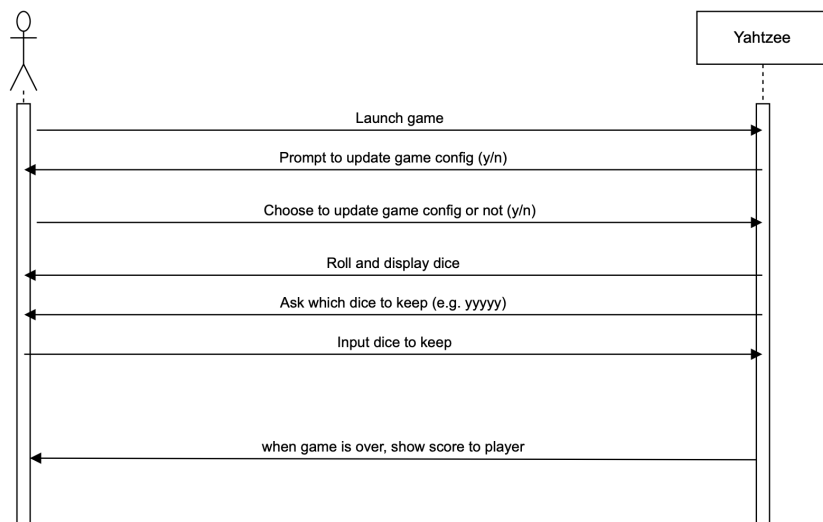
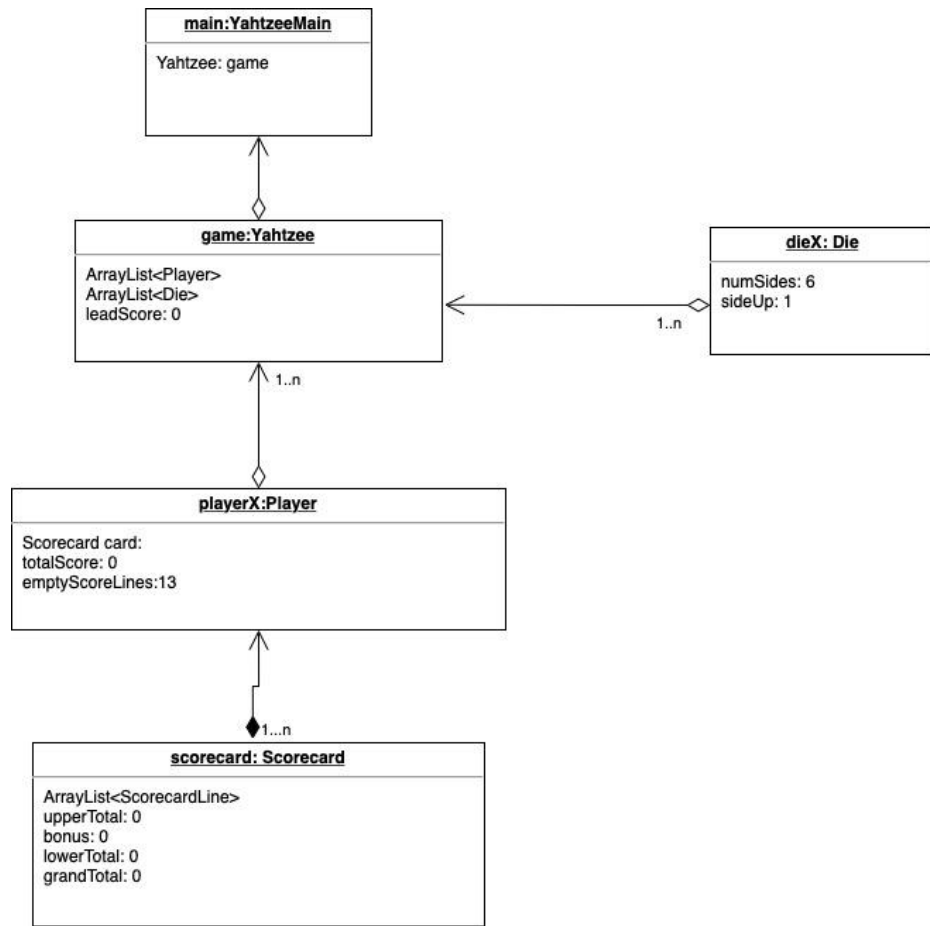
## Part 2: Project Functional Requirements

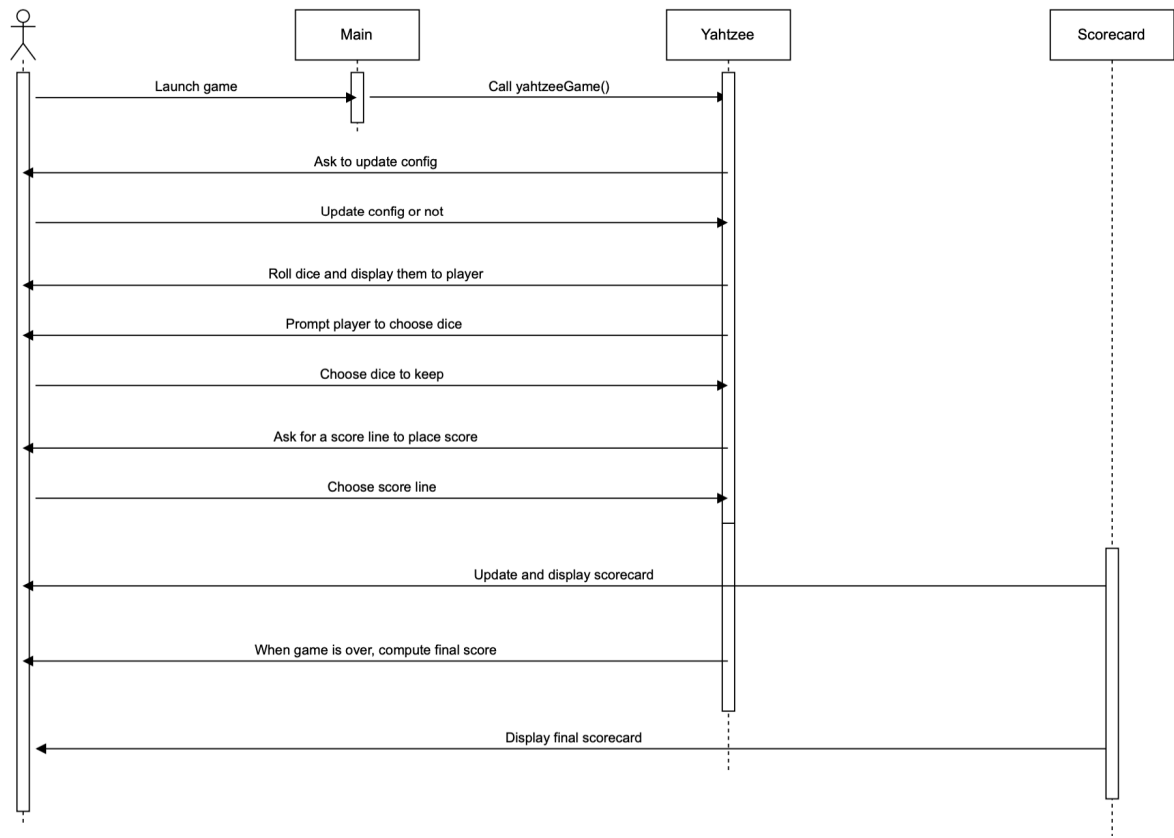
Prior ity	Purpose	Inputs/Needs	Operators/Acto rs	Outputs
High	The user shall be able to choose the number of players for the current game of Yahtzee	Number of players, need at least one, likely cap at some reasonable value to avoid excessively long games	The user and the main game loop	Number of players in the main game loop shall be set to the users value
High	The user shall be able to select an option at any point on their turn to	User must press a button on the GUI	Scorecard object and the user	The GUI must change to display the scorecard with accurate score

	view the current scorecard			values for the current game
High	As a part of the user's turn, they shall be able to roll the current hand of dice	User must press a button on the GUI, up to 3 times on a turn, only the "unkept" dice should be affected	Player/Hand/Dice objects and the user	The current hand of dice for the player must be updated to new values, and the GUI must change to reflect this
Low	User should be able to input a name for the player at the start of the game to identify them	User must enter a string for the name of the player, must be a valid string, should be capped at a certain number of characters. Add default value if player doesn't enter name	Player object and the user	The player object should be updated to hold this string, and the GUI should display this somewhere on screen
High	User should be able to choose dice to not be rolled on their turn, before rolling the dice	User must click on the corresponding widget on the GUI for the dice they would like to keep. Keeping all or none should be valid options	The user and the Player/Hand/Dice objects	The hand object should be updated with the dice that are not going to be rolled. Other objects that rely on these values should be updated/should reflect this change
High	After each set of 3 rolls, the play should select a score line from the remaining, unused scores. This score represents the score from the current hand of dice	User must be able to enter a value/click a corresponding widget on the GUI that correlates to a scoreline. Must be scoreline that has not already been selected	The scorecard/hand/dice objects are all involved as well as the user	The corresponding scoreline should be marked as used, and should now have a value equal to a score. Whatever objects use this value should also reflect this change
High	After a players turn has come to completion, the game should switch to the next player's turn	Player's turn should inform the main game loop that a turn has ended/A player should be prompted to end turn and should click a button. If only one player is in the game, they should just take	The game loop and player objects and the user	The current object that is being used to play the game must make sure it is using the correct player's information

		another turn. If the last player out of how many are participating ends turn, it should return to the first player's turn		
High	After all players have taken 13 turns, the amount of rounds in the game, the final scores should be determined	Needs to keep track of current turn of game/remaining scores to be taken	Main game loop and the player objects	Final scores should be updated for all players, and should be displayed/reflected somewhere on the GUI. Should also reflect any bonuses added
High	After the final scores have been calculated, a winner should be announced	The final scores must all be compared for each player	Player objects and the main game loop	The player that has the highest score should be stored somewhere, and the GUI should change and reflect the winners
Low	When displaying who won, players should be shown what places they got in the game	Need final scores for each of the players in the game	Player objects and the main game loop	The GUI should change and show the standing of players after the game is over
Low	When the game is first launched, a window should be launched that shows the Yahtzee logo and a button to begin the game	On launch, the window should be opened for the user. To move on from the screen, the user must click a button on the GUI	User and the main game loop	The game should progress to the setup for the game when the button is pressed

## Part 3: UML



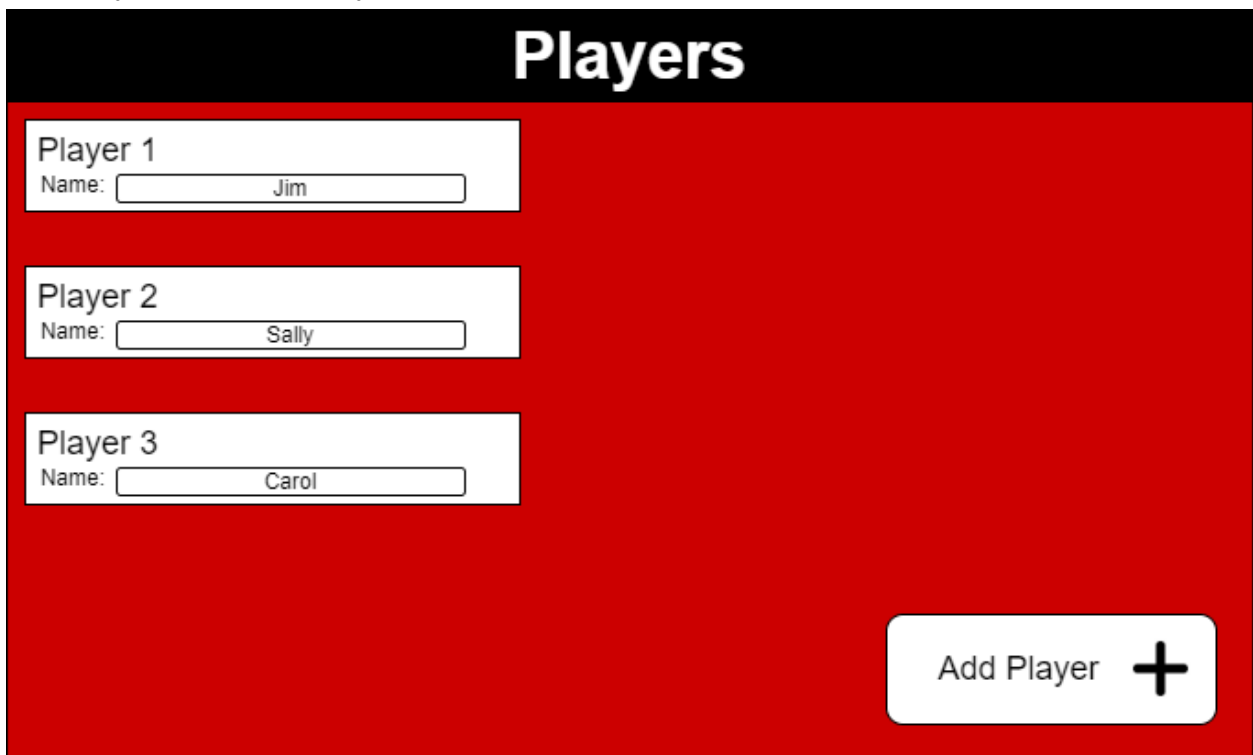


## Part 4: User Interface Mockup Design Sketch

Initial Interface



After Play button, Enter Players



The interface transitions to a screen for entering player names. It has a black header bar at the top with the word "Players" in a large, white, bold, sans-serif font. Below the header is a large red area. On the left side of the red area, there are three white rectangular boxes stacked vertically. Each box contains a label ("Player 1", "Player 2", "Player 3") and a text input field. The input fields contain the names "Jim", "Sally", and "Carol" respectively. In the bottom right corner of the red area, there is a white, rounded rectangular button with the text "Add Player" followed by a black plus sign (+).

First Roll, then Reroll

Yahtzee

Roll









Yahtzee

Roll






Select score after rolls are finished

Chose Score

Upper Section	Scoring
Aces  = 1	Count and add only Aces
Twos  = 2	Count and add only Twos
Threes  = 3	Count and add only Threes
Fours  = 4	Count and add only Fours
Fives  = 5	Count and add only Fives
Sixes  = 6	Count and add only Sixes
Total Score	
Bonus <small>If total score is 63 or over</small>	Score 35
Total of upper section	

Select the box of the score you would like to keep.









8




Lower Section	
3 of a kind	Add total of all dice
4 of a kind	Add total of all dice
Full House	Score 25
SM Straight <small>(sequence of 4)</small>	Score 30
LG Straight <small>(sequence of 5)</small>	Score 40
Yahtzee 5 of a kind	Score 50
Chance	Add total of all 5 dice
Yahtzee Bonus	✓ For each bonus
	Score 100 per ✓
Total of lower section	
Total of upper section	
Grand Total	

Done

## View Score Card

## Player 1's Current Score Card

Upper Section	Scoring	
Aces  = 1	Count and add only Aces	<input type="text"/>
Twos  = 2	Count and add only Twos	<input type="text"/>
Threes  = 3	Count and add only Threes	<input type="text"/>
Fours  = 4	Count and add only Fours	<input type="text"/>
Fives  = 5	Count and add only Fives	<input type="text"/>
Sixes  = 6	Count and add only Sixes	<input type="text"/>
Total Score		<input type="text"/>
Bonus <small>If total score is 63 or over</small>	Score 35	<input type="text"/>
Total of upper section		<input type="text"/>

Lower Section		
3 of a kind	Add total of all dice	<input type="text"/>
4 of a kind	Add total of all dice	<input type="text"/>
Full House	Score 25	<input type="text"/>
SM Straight <small>(sequence of 4)</small>	Score 30	<input type="text"/>
LG Straight <small>(sequence of 5)</small>	Score 40	<input type="text"/>
Yahtzee 5 of a kind	Score 50	<input type="text"/>
Chance	Add total of all 5 dice	<input type="text"/>
Yahtzee Bonus	✓ For each bonus	<input type="text"/>
	Score 100 per ✓	<input type="text"/>
Total of lower section		<input type="text"/>
Total of upper section		<input type="text"/>
Grand Total		<input type="text"/>

# Final Scores

Player 1: 112 Points

Player 2: 134 Points

Player 3: 78 Points

**Winner**

Player 2

## Part 5: Project Planning

Tasks:

- Part 2: (This assignment): 3 days - March 30th
  - UML design (Jesse)
  - UI design (Jackie and Katie)
  - Functional Requirements (John and Katie)
  - Summary (Jackie)
- Finish individual assignment (Everyone): 1 week - April 10th
- Look at each others code: 1-2 hours - April 11th
  - Break classes up
- Code (each person implements their own classes): 2 weeks - April 27th
- Testing (each person tests their own classes): 1 week - April 27th
- Create powerpoint (Everyone): 2 hours - May 3rd
- Final Report (Break up decided later): 4 hours - May 7th
- Peer Evals (Individual): 10 minutes - May 7th