Milestone 3 - Revised

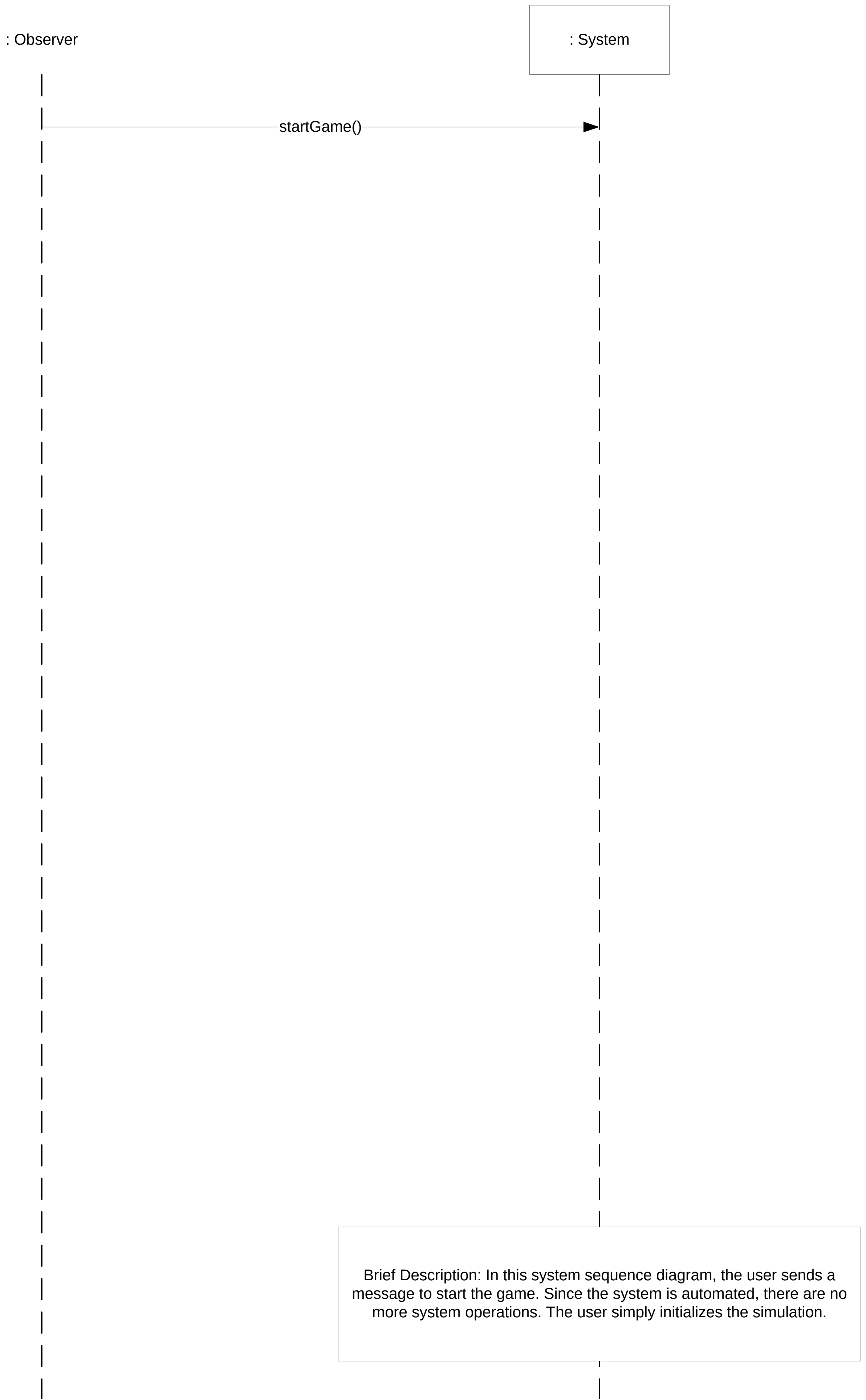
Concurrent Poker Player Team

Mark Jenne, Bennie Waters, Ian Roberts, Sarah Jabon

1/19/2010

Contents

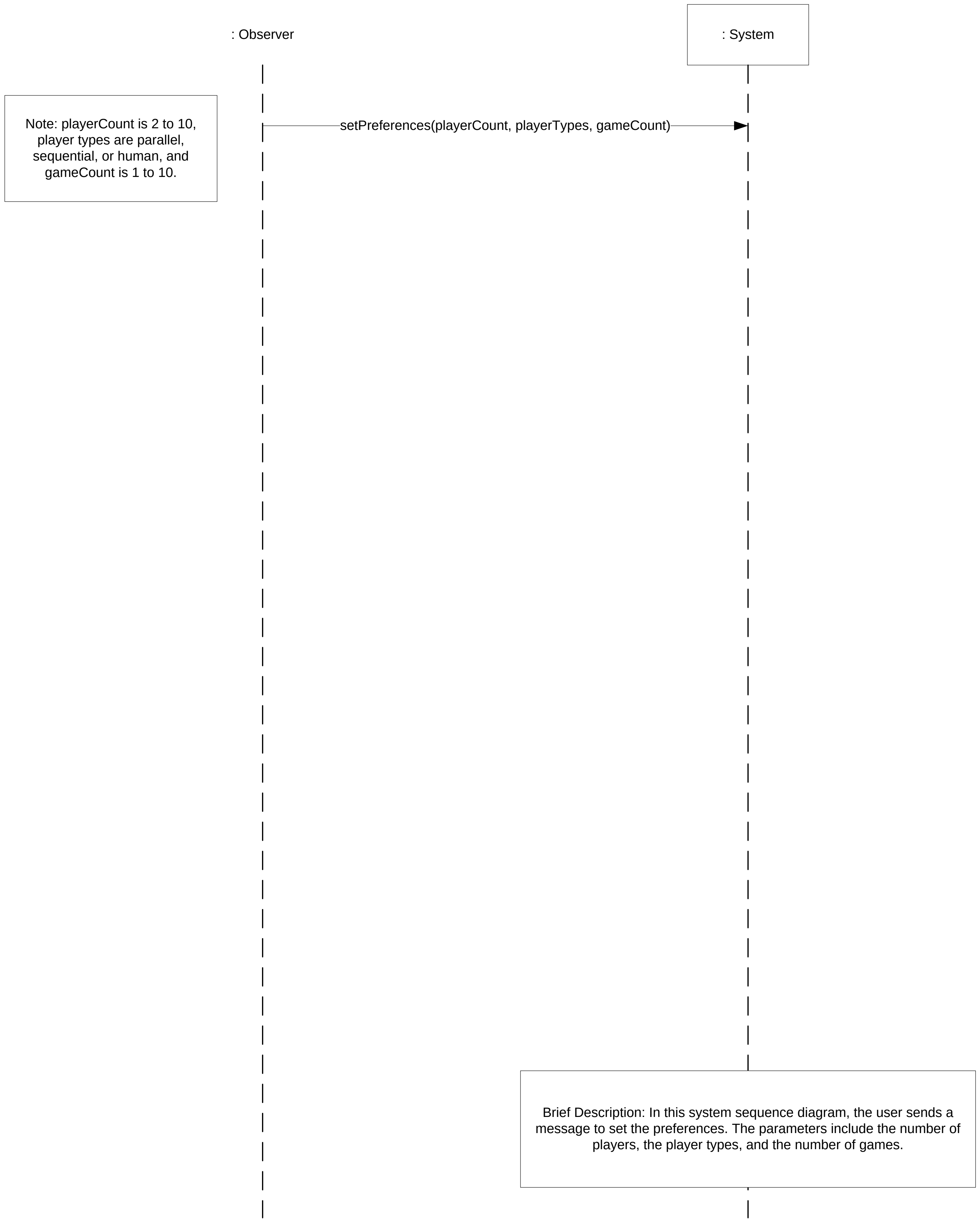
SSD – Use Case 1
(Start Game)



Note: The system is automated, so it does not require any more system operations.

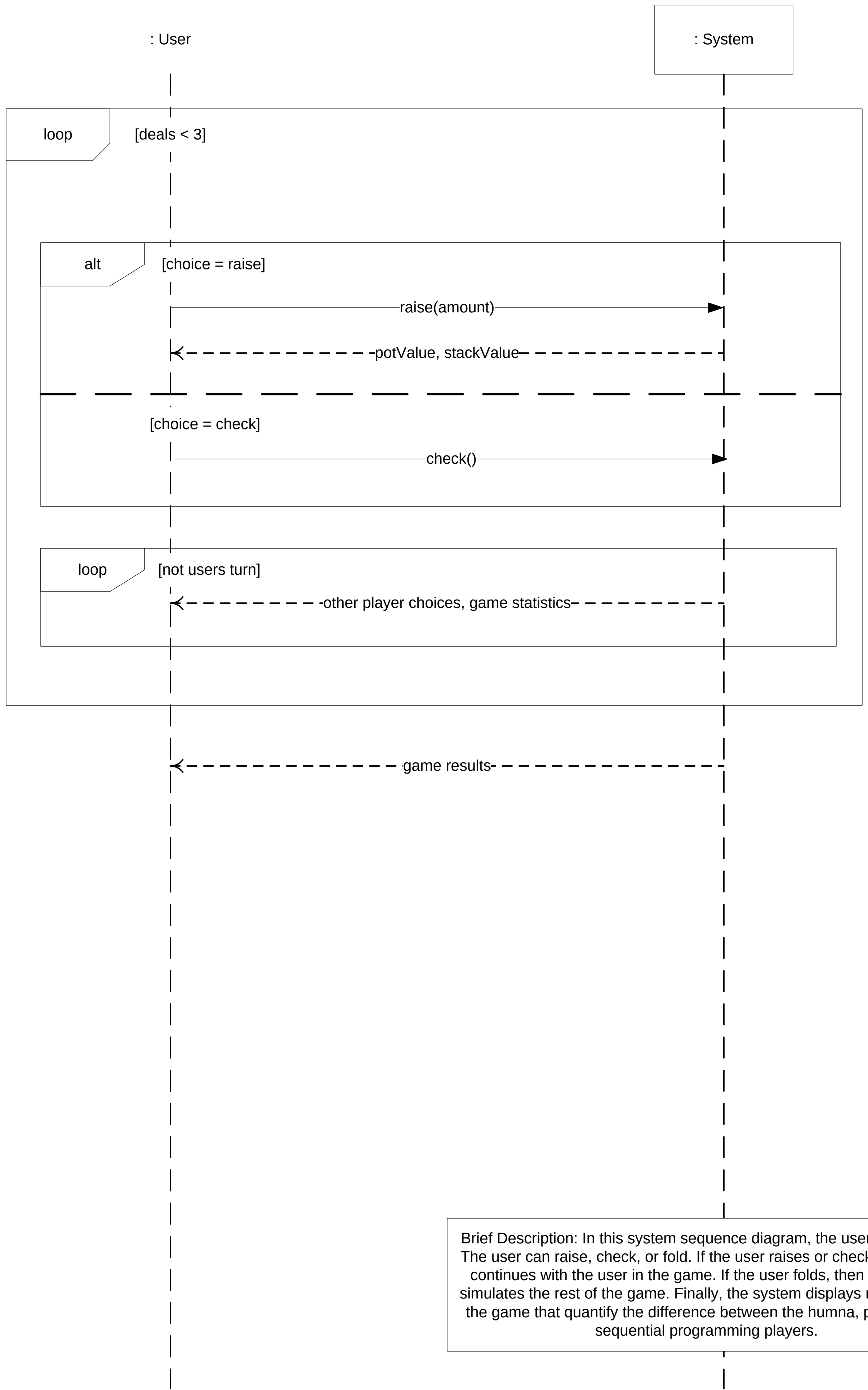
Brief Description: In this system sequence diagram, the user sends a message to start the game. Since the system is automated, there are no more system operations. The user simply initializes the simulation.

SSD – Use Case 2
(Set Preferences)


: Observer
: System
Note: playerCount is 2 to 10, player types are parallel, sequential, or human, and gameCount is 1 to 10.
setPreferences(playerCount, playerTypes, gameCount)

Brief Description: In this system sequence diagram, the user sends a message to set the preferences. The parameters include the number of players, the player types, and the number of games.

SSD – Use Case 3
(Take Turn)



Note: The user can choose to fold at any time. In this case, the system finishes the simulation without the user. Then, the system displays the game results.

Brief Description: In this system sequence diagram, the user take a turn. The user can raise, check, or fold. If the user raises or checks, the game continues with the user in the game. If the user folds, then the system simulates the rest of the game. Finally, the system displays results about the game that quantify the difference between the humna, parallel, and sequential programming players.

Contract CO1: startGame

Operation:	startGame()
Cross References:	Use Cases: Start Game
Preconditions:	The system is set to create a game with two players – one sequential and one parallel.
Postconditions:	An instance of Game, *game*, was created An instance of gameStatistics, *gameStats*, was created Attributes of *gameStats* were set based on simulation

Contract CO2: setPreferences

Operation:	setPreferences(playerCount, playerTypes{List}, gameCount)
Cross References:	Use Cases: Set Preferences
Preconditions:	The application has initialized properly Parameter menus are functioning properly An instance of Game, *game*, was created
Postconditions:	*game.playerCount* was set to playerCount *game.gameCount* was set to gameCount *player*, an instance of HumanPlayer, was created Instances of other Players were created corresponding to each playerType

Contract CO3: raise

Operation:	raise(amount)
Cross References:	Use Cases: Take Turn
Preconditions:	The game is in play It is the user's turn in the game An instance of Game, *game*, was created *player*, an instance of HumanPlayer, was created Instances of other Players were created
Postconditions:	*game.pot* was increased by amount *player.stack* was decreased by amount

Contract CO4: check

Operation:	check()
Cross References:	Use Cases: Take Turn
Preconditions:	The game is in play
	players{list} are playing game
	player[i] is currentPlayer
	player[i+1] is nextPlayer
Postconditions:	*player[i]*'s turn ended
	player[i+1]'s turn began

Contract CO5: fold

Operation:	fold()
Cross References:	Use Cases: Take Turn
Preconditions:	The game is in play
	players{list} are playing game
	player[i] is currentPlayer
	player[i+1] is nextPlayer
Postconditions:	*player[i]*'s turn ended
	player[i+1]'s turn began
	player[i] was removed from *players{list}*