CMSC 325 7980 Instructor: Armitava Karmaker
Home Work 1 Student: Dakin T. Werneburg

Question Analysis:

1. What is the task I am being asked to do?
2. Will this be command line or GUI interface?
3. What language should this be written in?
4. How many players should play this game?
5. How to determine the winner, i.e. place wager, best two out of three, etc.

Program Design:

1. Purpose of homework assignment: Create a basic coin-flip guessing game
2. Design Decisions and Assumptions: I chose to do a GUI interface using the Java Swing class and switch panels using CardLayout. I allowed the player to choose either the computer or another player, but will be limited to two players. I chose the winner to be the first to get to three correct guesses. Due to my limited knowledge in graphical programming, had to be very simple in my layout of the panels.

Test Plan:

Input	Expected Output	Actual Output	Passed or Failed
Intro view: select "OK"	Switch to main menu view	Switched to main menu view	Passed
Main Menu: Select the following: Player vs. Player, Player vs. Computer	Switch to coin selection and display correct current players and the correct score each turn. If playing computer, It should revert back to player one but computer score is updated.	View switches to coin selection and properly displays current player and their score.	Passed
Main Menu: Select "Simulate 100 Tosses"	Switch to simulate view, display back button, quit button, coin animation, and the results of 100 tosses	View switches to view display and properly displays all elements and each time Simulate 100 Tosses is clicked, a new set of results is displayed	Passed
Coin Selection: Select "Heads" or "Tails"	Switch to flip view with head side of coin and flip button displayed.	View switches to flip view and Head side of coin and flip button displayed	Passed
Flip view: Select "Flip"	Switch to coin animation view with the coin spinning and stop button displayed.	View is switch and coin spinning, with stop button displayed.	Passed
Coin Animation: Select "Stop"	Switch to results view and displays the coin that landed and the next player button	Spinning coin stopped and switched to results view, and correct button is displayed	Passed

Results view: Select "Next Player"	If Playing the computer, it should display the computers selection and the results of the toss and with a continue button If player another player, should switch back coin selection view and update scoreboard. If either player has a score of 3 it switches to end game	If playing computer, computer selection and actual results are displayed with a continue button. If playing another person, it switches back to coin selection and score and current player is properly viewed, and if any player gets a score of 3, the end game view is displayed.	Passed
End of Game view: Play Again or Quit	If Play again, scores reset and coin selection view is displayed, if quit, program quits.	If Play again, scores reset and coin selection view is displayed, if quit, program quits.	Passed

Error Handling:

IOException: used to verify if images are correctly loaded before beginning game is found during window initialization.

Lessons Learned:

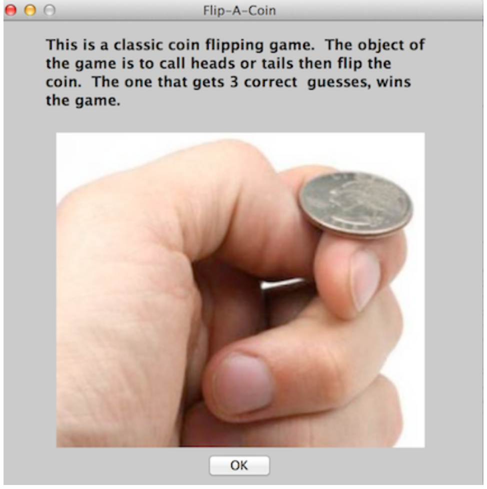
I think I took more on my plate than needed to be, as more obstacles were presented as my program grew. But I really like a challenge so there were many lessons learned. I will just cover the top ones.

1. It took many hours to learn about animation using paintComponent. I learned that animation is just loading a picture; adding another picture, then repaint the image.
2. I really had a hard time with variables when working with CardLayout and ActionListeners. I learned that once the method is called to display the panel that unless you call the method again, the variables don't change. I didn't know how to use cardlayout with different class. My original plan was to create a class for each of my cards.
3. Working with ImageIcon was simpler in adding images to labels but unfortunately you cannot check to see if the file is missing, or at least I don't know how. So, I learned to use BufferedImage and then add BufferedImage to my ImageIcon, then ImageIcon to my JLabel.

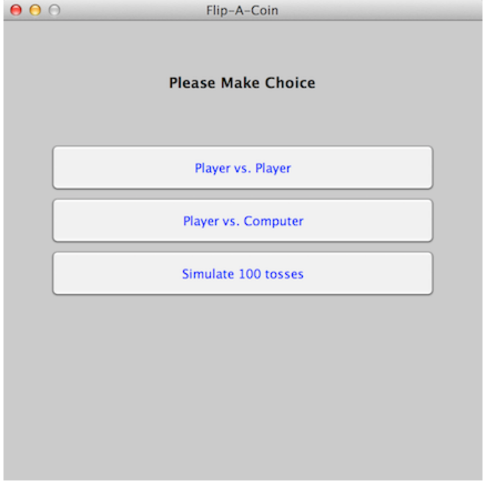
Possible Improvements:

1. Improve the animation so coin is going up and down while repainting
2. Improve the layout and feel of the GUI
3. Code to be broken down in to multiple classes, once I figure out how to use multiple classes and the use of CardLayout

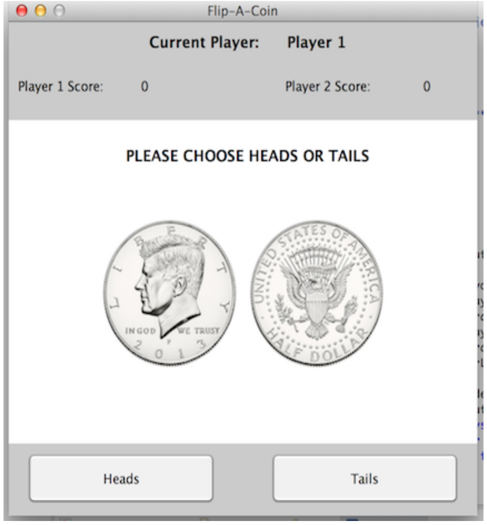
Screen Shots:



This is a classic coin flipping game. The object of the game is to call heads or tails then flip the coin. The one that gets 3 correct guesses, wins the game.



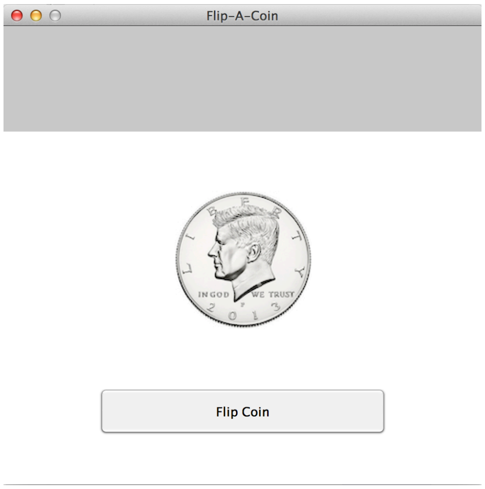
Please Make Choice

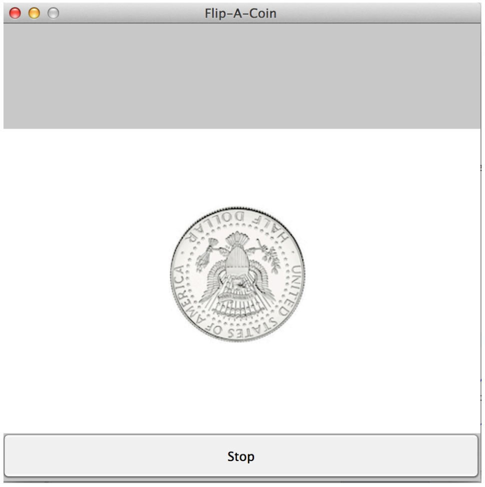


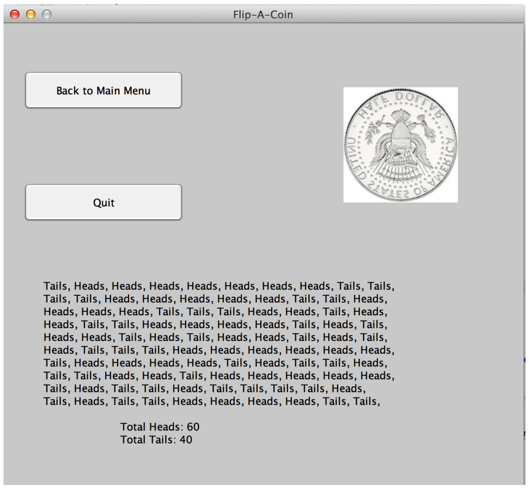
Current Player: Player 1

Player 1 Score: 0 Player 2 Score: 0

PLEASE CHOOSE HEADS OR TAILS







```
Tails, Heads, Heads, Heads, Heads, Heads, Heads, Heads, Tails, Tails,
Tails, Tails, Heads, Heads, Heads, Heads, Heads, Tails, Tails, Heads,
Heads, Heads, Heads, Tails, Tails, Tails, Heads, Heads, Tails, Heads,
Heads, Tails, Tails, Heads, Heads, Heads, Heads, Tails, Heads, Tails,
Heads, Heads, Tails, Heads, Tails, Heads, Heads, Tails, Heads, Tails,
Heads, Tails, Tails, Tails, Heads, Heads, Heads, Heads, Heads, Heads,
Tails, Heads, Heads, Heads, Heads, Tails, Heads, Tails, Tails, Heads,
Tails, Tails, Heads, Heads, Tails, Heads, Heads, Heads, Heads, Heads,
Tails, Heads, Tails, Tails, Heads, Tails, Tails, Tails, Tails, Heads,
Tails, Heads, Tails, Tails, Heads, Heads, Heads, Heads, Tails, Tails,

          Total Heads: 60
          Total Tails: 40
```