Cassandra Ziesman

Professor Rushing

C# Level I

17 May 2021

C# Final

Objective: For this project, you will be creating a High or Low Game using C#.

You will accomplish the objective by:

- Manipulating and Creating a GUI
- Importing Resources (graphics)
- Creating and Calling Methods
- Utilizing Event Handlers
- Using Random.Next Method to select a number from a number
- Use Switch statements to affect displays
- Parse to convert String to Integer
- Use If-Else Statements and Operators to Check Arguments.
- Create a Message Box to Display Text

Logic: High and Low is a game where two cards are placed with their back facing up. One card is flipped up to reveal the value. The player must guess if the card with the back facing up is higher or lower than the revealed card.

The major steps are to create the GUI which will show the cards with the back side down. A button to flip the first card to get display the flipped card and get the value. Then two buttons to choose to display the next card, grab that value, and then find out if it's higher or lower than the first card.

Materials: This game requires eleven total images located in the Instructions/Images.

Part 1: Use the Form Designer to Create the GUI Layout.

1. Use the Label tool to create the title with font size 16.5pt. Add another label below to include the instructions of the game.
2. Create the display for cards. Use a container and picture box. Make sure the picture boxes are labeled correctly.
    a. Tip: The images for the cards are size 203 x 303.
3. Create 2 buttons labeled High and Low. Make sure the buttons are named and disabled. Place these buttons in a column on the right side.
4. Create buttons for Play and Exit on the lower edge of the form.
5. Create two pairs of labels (four altogether) for the card values.
    a. Example: One pair of labels will have a label with the text: “Dealer’s Card:” and the other label having no text, fixed3d Border Style, and have AutoSize as False. The textless labels will be referred to as dealerLabel and userLabel in these instructions.
6. Click the triangle on the picture and import all 11 images from Instructions/Images in your resources. Use cardBack.png for both.
7. Adjust Form width and height if needed to look clean and make sure to properly label your form.
    a. Tip: The image above has a size of (600, 550).
8. Double-check to make sure that all things are properly named for the next part.

PART 2: Coding the game

1. Create the PlayGame() Method. Use the Random.Next Method to return a random integer between 1 and 10. Create a Switch statement with ten cases that utilize each

possible integer to display their coordinating card in the Dealer's picture box and display the integer in the Dealer's Value output label.

   a. Tip: To avoid using multiple picture boxes. You can use (pictureBox).Image = Properties.Resources.(image file).

2. Create the UserGuess() Method. Like the PlayGame() Method, use Random.Next Method, and a Switch statement to select a random number between 1-10 and use the switch statement to change the image and label text.
3. Go back to PlayGame() Method and add statements to 'reset' the User's card and label so that they'll display the cardBack graphic and erase any text in the label.
4. Double Click on the Play button to create an event. In that event call the PlayGame() method, enable High and Low buttons, and disable the Play button.
5. Click on the High button to create an event. Call the UserGuess(); Method and write code to disable the High and Low buttons, and enable the Play button. Declare two integer variables and have their values be the dealerLabel and the userLabel.

   a. TIP: Use the Parse method to convert from string to integer.

   Create an if-else statement that compares the values of the dealer's card and the user's card, and shows a MessageBox to state the outcome.

   b. TIP: Remember to use the correct operators and add conditions and outcomes if both cards have the same value.

6. Click on the Low button and use the code from the High button event as a guide. Make sure to call the method, disable/enable buttons, declare the variables and parse them.

Finally, create the if-else statement to check if the user's card is lower than the dealer's card.

7. Create the event to close out the application by clicking on the Exit button.

PART 3: Checking for Errors

- Make sure that there are no errors. Make sure that there are two methods. Each method uses Random.Next to select a number from 1-10. Have a Switch statement to display the corresponding card in the correct pictureBox and value in the correct label.
- Make sure each button has an event.
- Play button calls a method and enables/disables buttons.
- High and Low each calls a method, enables/disables buttons, parses a string value to an integer, and has an if-else statement to figure out if the user won.
- Make sure the Exit button successfully closes the Application.