**Written Plan**

This Rock Paper Scissor program contains 3 classes:

* PatelRockPaperScissor class: GUI runner main class that extends JFrame class, creates a new window, and places objects on the window using panels
* computerGenerate class: generates a random rock, paper, or scissor computer choice. Takes in user entered input and the random computer choice to compute the result that is displayed on the screen after user clicks play.
* checkInput class: checks if the user entered string is valid or invalid using exception handling. This class also inherits all the methods from computerGenerate class. In the main class, an object referencing checkInput class is created and it uses this object to access computerGenerate methods through inheritance.

**PatelRockPaperScissor Class:**

* public PatelRockPaperScissor() : Constructor that places objects on an empty window using panels
  + Sets the window attributes: size, layout (7 rows and 1 column), set visible, defaultCloseOperation
  + 7 panels are added to the window
* Panel 1 holds information and winning criteria that the player can read
* Panel 2 holds the user enter label, textfield, and image label that holds an image (rock, paper, or scissor)
* Panel 3 holds the play button
* Panel 4 holds the computer choice textfield that shows when the user clicks play. This field is is not visible to user before clicking play.
* Panel 5 holds result text field that displays if the user won, lost, or tied. This field is not visible to user before clicking play.
* Panel 6 holds play Again button to play the game again
* Panel 7 holds Quit button that closes the window
* Action Listeners:
  + private class CalcButtonListener: Executes when the user clicks play
    - Gets input from the user and sends it to checkInput() class constructor by creating an object. The checkInput class constructor calls computerGenerate parent class. checkInput class returns if the user entered a valid choice or not.
    - If the user did enter a valid choice…
      * then using playerNum method from super class computerGenerate, the program calculates if the user entered rock, paper, or scissor. Based on the return, the related image of rock, paper, or scissor is shown on the screen in panel 2.
      * Gets computer generated random choice by calling randNumGenerate method from super class and sets the returned string to computerChoice textField
        + Based on the random choice, the related image of rock, paper, or scissor is shown on the screen in panel 4.
      * The result is calculated by comparing user and computer choice by calling compare() method from the super class. This result is set to result textField
      * Both the result and computerChoice textFields are then set to be visible on screen.
      * User enter text field is set to non editable so that user cannot play the game unless the play again button is clicked
      * Screen is refreshed using revalidate() and repaint() method so that all the changes are visible on the screen
    - If the user entered text is invalid, then a pop up is shown in the screen asking the user to enter again.
* private class playAgainButtonListener: Executes when play again button is clicked
  + When this button is clicked, the user enter text field is set to editable and the previously entered user choice is erased.
  + The images are removed from the screen
  + Screen is refreshed using revalidate() and repaint() method so that all the changes such as removing the images and changing the attribute of user enter text field are visible on the screen
  + After this is clicked, the user can play the game again.
* private class quitButtonListener: Executes when quit button is clicked
  + When this button is clicked the window terminates and program stops
* Main method
  + Calls the RockPaperScissor() constructor

**computerGenerate class:**

* Creates a random class object called computerGenerate: Uses random class to generate computer choice rock, paper, or scissor
* Methods:
  + Constructor computerGenerate(String userInput):
    - sets the userInput string to input variable
  + int playerNum(): returns player variable 1, 2, or 3
    - Converts the userInput to integer values
      * If the user enters rock, player variable is set to 1
      * If the user enters paper, the player variable is set to 2
      * If the user enters scissor, the player variable is set to 3
  + String randNumGenerate(): returns computer choice
    - Generates a random number between 1 and 3
    - Converts each integer value to string using conditional statements:
      * If computer generates 1, the method returns “Rock”
      * If computer generates 2, the method returns “Paper”
      * If computer generates 3, the method returns “Scissor”
  + Int getCompChoice(): returns computer generated integer value 1, 2, or 3
  + String compare(): returns the result by comparing the values of computer and user
    - Example: if player choice is 1 (rock) and computer choice is 3 (scissor), the method returns “You win!! Rock smashes Scissor”
    - If all conditions fail, the program returns “its a tie”

**checkInput class():**

* Methods:
  + Constructor checkInput(String input)
    - Calls the super constructor and passes input as the parameter
    - Sets input string to userInput String
  + Boolean check(): returns true if the input is valid. Returns false if input is invalid
    - Evaluates if the string entered is valid or not through try and catch statement
    - The only valid inputs this method excepts are Rock, Paper, Scissor. Any other input throws an IllegalArgumentException

One of the main methods for testing used is by running the code after every block.

**Testing Plan:**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Input** | **Output** |
| Testing that when the user enters Rock 10 times, different computer choices are generated. | Rock | 1. Computer Chose Rock 2. Computer Chose Scissor 3. Computer Chose Rock 4. Computer Chose Rock 5. Computer Chose Scissor 6. Computer Chose Scissor 7. Computer Chose Scissor 8. Computer Chose Rock 9. Computer Chose Paper 10. Computer Chose Paper |
| Testing that when the user enters Paper 10 times, different computer choices are generated. | Paper | 1. Computer Chose Scissor 2. Computer Chose Rock 3. Computer Chose Scissor 4. Computer Chose Paper 5. Computer Chose Paper 6. Computer Chose Paper 7. Computer Chose Paper 8. Computer Chose Rock 9. Computer Chose Rock 10. Computer Chose Rock |
| Testing that when the user enters Scissor10 times, different computer choices are generated. | Scissor | 1. Computer Chose Rock 2. Computer Chose Paper 3. Computer Chose Scissor 4. Computer Chose Rock 5. Computer Chose Scissor 6. Computer Chose Rock 7. Computer Chose Scissor 8. Computer Chose Scissor 9. Computer Chose Scissor 10. Computer Chose Paper |
| Testing that when the user enters Rock, rock icon shows up | Rock |  |
| Testing that when the user enters paper, paper icon shows up | Paper |  |
| Testing that when the user enters Scissor, scissors icon shows up | Scissor |  |
| Testing that when computer generates Rock, rock icon shows up | Rock |  |
| Testing that when computer generates Paper, rock icon shows up | Rock |  |
| Testing that when computer generates Scissor, rock icon shows up | Rock |  |
| Testing that when I enter lower case rock, the program considers it as valid input | rock |  |
| Testing that when I enter lower case paper, the program considers it as valid input | paper |  |
| Testing that when I enter lower case scissor, the program considers it as valid input | scissor |  |
| Testing that when I enter anything other than rock, paper, or scissor, the program gives an error | Roc |  |
| Testing that when I enter Rock and computer enters rock, the result is a tie | Rock |  |
| Testing that when I enter Rock and computer enters paper, the result is the computer wins | Rock |  |
| Testing that when I enter Rock and computer enters scissor, the result is the user wins | Rock |  |
| Testing that when I enter Paper and computer enters Paper, the result is a tie | Paper |  |
| Testing that when I enter Paper and computer enters Rock, the result is the user wins | Paper |  |
| Testing that when I enter Paper and computer enters Scissor, the result is the computer wins | Paper |  |
| Testing that when I enter Scissor and computer enters scissor, the result is a tie | scissor |  |
| Testing that when I enter Scissor and computer enters Rock, the result is the computer wins | scissor |  |
| Testing that when I enter Scissor and computer enters paper , the result is the user wins | scissor |  |
| Testing that when the user clicks play, user cannot enter another choice | Rock | Before:  After: |
| Testing that when the user clicks play again, user cannot click play button again | rock | Before:  After: |
| Testing that when the user clicks play, the computer choice and the result appears when you first play the game. | rock | Before:  After: |
| Testing that when the user clicks play again button, the user can enter another choice | -- | Before:  After: |
| Testing that when the user clicks play Again button, the images disappear | -- | Before:  After: |
| Testing that when user clicks exit, the window closes and the program stops executing | -- | Works |