RPSLS User Stories

**Out of 65 points**

Using the concepts of OOP by creating classes and using objects (instances of those classes) to interact with each other, create a console version of the classic game Rock Paper Scissors Lizard Spock.

**User stories:**

**(5 points): As a developer, I want to make good, consistent commits.**

**(15 points): As a developer, I want to find a way to properly incorporate inheritance into my game.**

**(5 points): As a developer, I want to account for and handle bad user input, ensuring that any user input is validated and reobtained if necessary.**

**(10 points):**As a developer, I want to store all of the gesture options/choices in a List<T>. I want to find a way to utilize the list of gestures within my code (display gesture options, assign player a gesture, etc).

**(10 points): As a player, I want the correct player to win a given round based on the choices\* made by each player.**

**(10 points): As a player, I want the game of RPSLS to be at minimum a ‘best of three’ to decide a winner.**

**(10 points): As a player, I want the option of a single player (human vs AI) or a multiplayer (human vs human) game.**

\*

Rock crushes Scissors    
Scissors cuts Paper   
Paper covers Rock   
Rock crushes Lizard   
Lizard poisons Spock   
Spock smashes Scissors   
Scissors decapitates Lizard   
Lizard eats Paper   
Paper disproves Spock   
Spock vaporizes Rock

<https://www.youtube.com/watch?v=Kov2G0GouBw>

Classes:

Program Class

Game Class

Player Class (Abstract)

Human Class

Computer Class

~~Gestures Glass~~

1. Write out the steps to play RPSLS
2. What classes, what variables, what methods. This is the skeleton.
3. This is where the coding happens.

Write out Steps

1. Display rules.
2. Choose Player **How many players?**
   1. **Player vs Player**
   2. **Player vs Computer**
3. Name or Player1 vs Player2
4. Display Gestures
5. Choose Gesture
6. Compare Gesture
   1. Tie – if tie loop back to number 4
7. Determine Round Winner
8. Increment Score
9. Check if Game Winner
   1. If yes, ask play again
   2. If no, loop back to number 4.

Classes –

Game

Player player1;

Player player2;

List<string> gestures

**(want to change this private)** public string GetNumberOfPlayers()

{  
CW(“1 or 2 players?”);

string numberPlayers = CR();

return numberPlayers;

}

**(want to change this private)** public void SetPlayers(string numberofPlayers)

{  
if (numberofPlayers == “1”)

{player1 = new Human();

player2 = new AI();  
}

else if(numberofPlayers == “2”)

{player1 = new Human();

player2 = new Human ();

}

player1 = new AI();

player2 = new AI();

Public void RunGame()

{

string players = GetNumberOfPlayers();

SetPlayers(players;)

player1.ChooseGesture();

player2.ChooseGesture();  
}

Player player1 = new Human(); **This is how to instantiate a subclass**

Player (abstract) you will never be able to instantiate a Player at any point, so you will need to instantiate the sub classes.

public string name;

public int score;

public string gesture;

public abstract void ChooseGesture();

probably need and abstract void ChooseName() method as well.

Human : Player

public override void ChooseGesture()

{  
Console.WriteLine”Choose Gesture:”

gesture = Console.Readline();

}

AI : Player

public override void ChooseGesture()

{

// Random function

}