Web Development – Mr. Turner

Project – XTreme Rock Paper Scissor

<u>Project Overview</u>

XRPS is a two player game where each player controls a limited arsenal of rocks, papers, and scissors that they must play against each other. The object of the game is to capture all of the weapons in your opponent's arsenal.

A human will play against the computer.

During a round, each player will choose to play a weapon from his or her arsenal (a rock, a paper, or a scissor). The winner of the round is determined in the following way:

- Rock beats scissor
- Scissor beats paper
- Paper beats rock

The winner of the round collects the opponent's played weapon and adds it to their arsenal. In the event of a tie, each player recovers their own weapon

If I play rock and my opponent plays scissor, I win the round and add the scissor to my arsenal.

The game ends when one player has all of the weapons.

<u>Display</u>

The list below includes the essential elements of the page.

- Use HTML and CSS to create an intuitive interface.
- Your page must include a section for each player.
    - Each section must contain 3 images, one for each type of weapon.
    - Each section must contain an indicator of how many of each weapon is remaining in the arsenal.
- A graphical display showing which weapon each player plays each round.
- A message area.
- Your page must have a log of rounds.
    - The log must show what was played by each player and who won the round.
- Include a rules section or a rules page.
- Include a reset button for restarting the game.

<u>Functionality</u>

Each player begins with 5 of each weapon.

In order to play a weapon, the player should be able to click on the appropriate image.

The computer will choose its weapon at random.

Remember that neither player may choose a weapon if it has exhausted its supply.

If a player plays a weapon more than 3 times in a row, that weapon “breaks”. It is removed from the game.

After each player plays a weapon, the system will show them on screen and flash a message indicating who won. Then it will update the log. There will be an indication on screen if a weapon breaks.

The player will drive each round simply by choosing a weapon.

When the game ends, show a message indicating the winner. Afterwards, the player can only continue by resetting the game.

The reset button will return all of the weapons back to the players and clear the log.

<u>Enhancements</u>

- AI - Give the computer some measure of intelligence. Try to incorporate one or more of the following ideas:
  - The computer should recognize when the player is out of a certain weapon and never play the weapon that would lose to it.
  - The computer will try to empty a player’s arsenal of a single weapon by always playing the one that beats it.
  - The computer will play defensively, using the weapon of which it has the most.
  - There should always be a chance that the computer will choose at random. This will prevent the user from catching on to any pattern.
  - The computer will never play a weapon more than 3 times in a row.
  - The computer will recognize when a weapon has been played 3 times in a row and play accordingly.
- Builders - The players can build new weapons.
  - A player can sacrifice any 2 different weapons to build one of the 3rd (a rock and a scissor to build paper, etc..).
  - The player should have buttons that allow him or her to do this.
  - The computer will automatically do it if it runs out of a weapon.
  - Remember that a player must have the appropriate resources in order to build.
- Tie Breakers - If the players tie over 3 rounds consecutively, each of them loses a weapon at random.
- Bigger Games - Create an interface whereby the user can change the number of weapons in the starting arsenal.
- Lost Cause - Make the program understand that a player with only a single type of weapon cannot win the game. End the game at that point.
  - Move all of the loser’s remaining weapons to the winner’s arsenal and flash a message explaining what happened.

    - Update the log appropriately.

<u>Necessary Programming Skills</u>

- Comprehension of the specifications sheet.
- Planning
    - Figure out the information you need to keep track of.
        - This information will become your global variables.
    - Plan out the individual tasks your program must perform.
        - Think through the steps for each task.
        - Think through the information your task needs (where does it come from?).
        - These will become your functions.
    - Plan out the user interface.
        - You can start with the barest interface, but you should have an idea what you want the final product to look like.
- Managing your variables
    - What's global, local, and passed through as parameters (hint - this program can use all three)?
    - Do you have a complete back end design (variables and functions that work the program)?
    - Does your back end inform your display?
- Assigning functions as tasks
    - Does your program sort out the different tasks into their own functions?
- Sequencing
    - Does your program sequence from the user interaction into the necessary functions?
    - Is there an efficiency to your code that flows from the design document?
- An intuitive user experience
    - Is your display appropriate to the program (what's viewable and what scrolling has to be done)?
    - Is your display adaptable to other resolutions?
    - Is the interface intuitive?