Homework Assignment 03 **Assigned:** Fri 14 FEB 2025
Rock, Paper Scissors, Lizard, Spock **Due:** Sun 23 MAR 2025

Instructions:

- The assignment is to be uploaded to the course repository (GitHub) by the due date, which is scheduled for 11:59pm ET that day since solutions will be distributed soon after.
- We expect that you will study with friends and often work out problem solutions together, but *you must write up your own solutions, in your own words*. **Cheating will not be tolerated.** Professors and TAs will be available to answer questions but will not do your homework for you. One of our course goals is to teach you how to think on your own and solve your own problems using your resources.
- We require that all homework submissions be neat and organized. **There will be point deductions if the submission is not neat** (is disordered, difficult to read, etc.).

Problem 1 [85 points]: Building a Network Game

Using the HW3 files in the ClassRepo, create a Linux C Socket game application called "spock" that plays a game of the form of the classic "Rock, Paper, Scissors" variety, but is modified using the rules highlighted in the television series, *The Big Bang Theory*. For a more detailed description of the play of the game, please reference the fandom site of *The Big Bang Theory* are the following URL:
https://bigbangtheory.fandom.com/wiki/Rock,_Paper,_Scissors,_Lizard,_Spock

To create your game application, you will need to embed code that allows the application to run either in client or server mode, toggled on the command line via the use of -c for client or -s for server mode. Your application will communicate across the network using TCP/IP socket programming to configure communication to and between the other application(s) running. One application must run in server mode and be able to advertise its available port for the game. The client must be able to take on the command line or as an entry field in the application the port of the server to which the game application should connect. Ultimately, one computer player will launch as the server, and the other computer player will launch as a client connecting to the server.

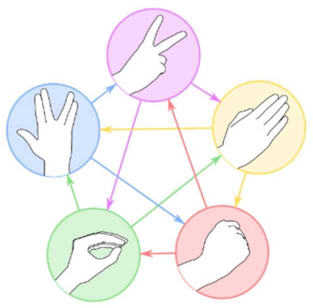
You must determine what the user interface will be for this game. This is entirely your choice. However, the more descriptive the user interface is, the better the overall score will be for this assignment.

The server and client need to identify who is player 1 and player 2. That negotiation is up to you.

R=rock; P=paper; S=scissors; L=lizard; K=Spock are the keys that represent the options of play when called to do so. Q=quit is used to quit the game. M=score is used to display the current score of the game. Score begins with each player having 0 points. A winning game achieves 1 point. A tied game achieves no points. T=reset can be entered by either player to reset the game score between the two players.

Game play is as follows: At a given three-second timeline, the players will send to each other their selection. Each player will receive the other players' move and then compare each others' selections to determine who wins. The scoring for the game is as follows:

Paper wins over Rock
Paper wins over Spock
Scissors wins over Paper
Scissors wins over Lizard
Spock wins over Scissors
Spock wins over Rock
Rock wins over Scissors
Rock wins over Lizard
Lizard wins over Spock
Lizard wins over Paper



The determination of the winner must be confirmed unanimously between the players. Once confirmed, the winner is declared and highlighted on the screen.

An optional game counter can be introduced to allow the players to track their overall game strategy in determining who is winning in a streak.

If the game is declared over and Quit is selected by any party, then the game applications should all gracefully exit the program and the ports released.

How to Begin

Much of the code used in the speak and speakd application of HW1 will be the essential parts of the application code that you can draw from to make the communication protocols work. The messages to be transferred between the application client(s) and the application server can be formatted in any reasonably efficient manner you find convenient. Be sure to document your communication protocols occurring between the server and client applications as this information will be important in a future assignment.

Problem 2 [15 points]: Enhancing the Network Game

Once the game works for two players (B-level success), the game should be modified to work for three, four, or five players (your choice, A-level success). Three players is all that is minimally required, but the game is designed to support up to five players since there are five options for play.

Grading

This assignment will have certain weighting associated with it. Your attempt to complete aspects will place you in line for greater reward at increased risk. To earn a grade of:

B: complete the assignment Problem 1 as stated above. Working code will solidify the B grade.

A: modify the code to support more than two players (3, 4, or 5 players). Working code will solidify the A grade.

Submission

You must submit a “makefile” to compile and link your programs starting from the source files only. DO NOT MAIL OBJECT CODE. Please call your application’s executable program “spock” and include instructions in a README file as to how to run the features of your application.

Ideally your code will run on the Khoury Linux Server in addition to your local computer, thus ensuring that anyone can play your game through a simple login to the Khoury Linux Server.

All files should be placed into a folder called HW3 and included in your GitHub private repository.