Program 1: ***Submit via file attachment with email***

For this question you will submit a file called charFreq.m as well as the sample output specified below as results.txt, with extra credit answered below.

Write a function that analyzes how many times the characters ‘a’ to ‘z’ occur in a file, and writes those counts, as well as their frequencies, to a specified output file.

```
function [ ] = charFreq( inputFile, outputFile )
 %
 % inputFile specifies the file whose characters should
 % be analyzed.
 %
 % outputFile will contain the counts of the characters
 % ‘a’ to ‘z’ found in the input file, and will
 % give the frequency of each character.
 % Frequency is the number of occurrences
 % divided by the total characters in the file.
 %
 % If inputFile or outputFile cannot be opened, the
 % function will print a descriptive error message and
 % return.
 %
```

Be sure to test your function on a test file. An example of the output can be downloaded on the course schedule.

Now, download the file “warpeace.txt” and run your function, generating a file called “results.txt”. Submit this results file with your code. You can find this file on the course schedule on the exam date.

Program 2: ***Submit via file attachment with email***

For this question you will submit one file, RPS.m, with one main function and three subfunctions.

Rock-Paper-Scissors is a two-player game that proceeds by rounds, and in each round both players decide to play one of the three titular moves: Rock, Paper, or Scissors. If the players play the same move then neither wins that round. However, if different moves are played, then scissors beats paper, paper beats rock, and rock beats scissors.

1) Write a subfunction that randomly picks one of rock, paper, or scissors and returns an ‘R’, ‘P’, or ‘S’ accordingly. Note that capitalization is important, because MATLAB does not consider ‘P’ to be the same as ‘p’, or ‘R’ to be ‘r’, or ‘S’ to be ‘s’.

```
function [ pick ] = RPSplay( pR, pP, pS )
% Input:
% pR is the probability (0-1) of picking rock
% pP is the probability (0-1) of picking paper
% pS is the probability (0-1) of picking scissors
% Output:
% pick is one of ‘R’ ‘P’ or ‘S’ denoting which move
is picked
```

2) Write a subfunction that plays one round of Rock-Paper-Scissors. This function takes six arguments that specify the probability of each player’s moves.

```
function [ winner ] = RPSround ( p1R, p1P, p1S
                                 p2R, p2P, p2S )
% Input:
% p1R, p1P, p1S are the probabilities of player 1
% p2R, p2P, p2S are the probabilities of player 2
% Output:
% winner is a scalar value indicating the winner
%        0 if neither player wins
%        1 if player 1 wins
%        2 if player 2 wins
```

3) The third subfunction of the file should play an entire game of rock-paper-scissors and report the winner. This should call the RPSround() subfunction repeatedly until one player wins the number of times specified by roundsToWin

```
function [winner plays] = RPSgame ( p1R, p1P, p1S,
                                    p2R, p2P, p2S,
                                    roundsToWin )
% Input:
%
% p1R, p1P, p1S are the probabilities of player 1
% p2R, p2P, p2S are the probabilities of player 2
% roundsToWin is the number of rounds needed for a
%    a player to win. Example: if roundsToWin=3,
%    then the game ends when one player wins a
%    total of three times.
%
% Output:
%
% winner is a scalar value indicating the winner
%       1 if player 1 wins
%       2 if player 2 wins
%
% plays is a scalar value saying how many rounds
%   were played before the game ends. For example,
%   if roundsToWin=3, then player 1 might have won
%   three times, player 2 might have won twice, and
%   they may have tied once, for a total of 6 plays
```

4) The main function of the file should play a specified number of games of rock-paper-scissors, print the overall winner, their win percentage, and average number of plays. For example, the invocation:

RPS(1/3, 1/3, 1/3, 1/3, 1/3, 1/3, 3, 10000)

might print:

Player 2 won 52% of the time with an average game length of 4.5 plays.

If roundsToWin is not specified, it should default to 3.
If totalGames is not specified it should default to 1000.

```
function [] = RPS( p1R, p1P, p1S,
                   p2R, p2P, p2S,
                   roundsToWin, totalGames)
% Input:
%
% p1R, p1P, p1S are the probabilities of player 1
% p2R, p2P, p2S are the probabilities of player 2
%
% roundsToWin is the number of rounds needed for a
%    a player to win. Example: if roundsToWin=3,
%    then the game ends when one player wins a
%    total of three times. Defaults to 3.
%
% totalGames is the number of games to play.
%    Defaults to 1000.
%
%
% Output: text only
```

Program 2 Questions (10 points): Submit in class

1) Play 1000 games of rock-paper-scissors for three rounds to win where each player picks rock, paper, and scissors evenly (i.e. with probability 0.333). Give the output of your program. Does your output make sense?

2) Play 1000 games for three rounds to win where player one picks moves evenly (i.e. with probability 0.333) and player two only plays rock and paper with probability 0.5. Give the output.

3) Play 1000 games for three rounds to win where player one picks moves evenly (i.e. with probability 0.333) and player two only plays rock with probability 1. Give the output.

4) Play around with other possible player strategies. Assume your opponent always picks moves evenly and at random. What does your investigation show?

Extra Credit (5 points)

Program 1 can be used to decrypt the following message, which was encoded with a Caesar Cipher (look it up online). Compare the letter frequencies in the message below with the frequencies you found in the longer work. Write a short description of your decryption process.

TAI XAZS PA KAG
IMZF FTQEQ YQEEMSQE
FA DQYMUZ EQODQF?

U IMZF FTQY FA DQYMUZ
EQODQF RAD ME XAZS ME
YQZ MDQ OMBMNXQ AR QHUX.

Program hints:

Program 1:

Remember that you can use the ASCII encoding to treat characters like numbers when it is convenient. For example:

- ‘a’ is ASCII 97
- ‘a’ + 1 is equal to ‘b’
- ‘a’ + 2 is equal to ‘c’
- Etc…

The file warpeace.txt has been converted entirely to lowercase. You don’t need to consider capitalization.

Program 2:

Be careful when comparing character values, ‘P’ is not the same as ‘p’.

Remember that only the top level function can be called from outside the function file.

Test each subfunction you create as you go. Make sure your results make sense before working on the next part of the program.