Assignment #4 (40 points; due 4/23/2018 at 11:59 P.M.)

For all Programs:

To get in the habit of writing pseudocode write the simple pseudocode for these programs. Put your pseudocode in comments at the top of your programs, as well as many lines within the code necessary to describe what's going on in the program. PSEUDOCODE IS REQUIRED FOR ALL PROGRAMS.

For each program make sure and include the following comments at the top (do this on all homework assignments from now on – this is required):

```
//Your Name
//CS 1, Section #ABC
//Assignment #X, Problem #Y
//Summary of the program
```

These programs address the basics from Chapters 6.

Perfect Number Finder

1. (20 points) Write a program that asks the user for input for two integer values: startval, and endval. The program will then find all Perfect Numbers that fall between startval and endval.

   A perfect number is a number equal to the sum of all its proper divisors (divisors smaller than the number) including 1. The number 6 is the smallest perfect number; because it is the sum of its divisors 1, 2, and 3 (1+2+3 = 6). The next perfect number is 28 (28 = 1+2+4+7+14).

   The search for the Perfect Number will require two loops (nested). The outer loop will step thru the range of numbers between startval and endval. The inner loop will step thru the values from 1 to (outerval/2) to determine the factors of the number.

   Write and use a function called isAFactor that accepts two int args, and determines if the second number is a factor of the first. It should return a bool value.

   When a number is found, print out a message: **Xxx is a Perfect Number.**

   If no Perfect Numbers can be found in the given range, print the message:

No Perfect Numbers found between xxxx and yyyy.

<u>Rock, Paper, Scissors</u>

2. (20 points) Write a program that lets the user play the game of Rock, Paper, Scissors against the computer. The program should work as follows:

    a. When the program begins, the user enters his or her choice of "rock", "paper", or "scissors" at the keyboard using a menu in a function, userChoice, that accepts no arguments and returns a character.

    b. Next, there should be a function, computerChoice, that accepts no arguments and generates the computer's play. A random number in the range of 1 through 3 is generated. (NOTE: You'll have to do some research on how to create random numbers correctly.) If the number is 1, then the computer has chosen rock. If the number is 2, then the computer has chosen paper. If the number is 3, then the computer has chosen scissors. The computer's choice is returned as a character.

    c. After, a function, determineWinner, will determine the winner between the user's choice vs. the computer's choice. (NOTE: It will return nothing, but it will take in the user's choice and the computer's choice as the two arguments.) The result is selected according to the following rules:

        i. If one player chooses rock and the other player chooses scissors, then rock wins. (The rock smashes the scissors.)

        ii. If one player chooses scissors and the other player chooses paper, then scissors wins. (Scissors cuts paper.)

        iii. If one player chooses paper and the other player chooses rock, then paper wins. (Paper wraps rock.)

        iv. If both players make the same choice, the game ends in a draw and there is no winner.

    d. Finally, after a result is selected, there should be a function, playAgain, in which the player should have the option of playing again. This should take no arguments and return a boolean.

Be sure that the program contains at least the four functions mentioned in parts a – d..