Recursion Coding Exercises

Sometimes the same solution can work both using loops and recursion. In completing these exercises, you are NOT allowed to use any type of loop. You must find a recursive method to do this. You can add any helper functions you need to support the recursive approach you want to take. Remember, the functions provided are useful for users, but some may need a second function to support recursion.

#1 File System Querying

File systems are naturally recursive structures that lend to recursive algorithms. We will work on this in class, so look at recordings to help you get started!

- Count the number of subdirectories in a folder
- Count the number of files in a folder
- Count the number of hidden files in a folder
- Compute the total number of bytes stored within a folder

#2 String Looping (35 points)

Programmers also use loops to process and query text. This exercise includes various examples of working with text and includes a starter video to get you started.

- Swap the case of the text - every upper case letter (A) is lower (a) and vice versa - **(5 points)**
  - `Tony Lowe` -> `tONY lOWE`
- Count the number of times a character appears in a block of text - **(5 points)**
  - `Tony Lowe` contains 2 'o's
- Strip any characters from the text that are not of the core alphabet (a-z, A-Z) - **(5 points)**
  - `Coder 4 ever!` -> `Coderever`
- Reverse the order of the text - **(5 points)**
  - `Tony Lowe` -> `ewoL ynoT`
- Determine if a word is a palindrome - **(5 points)**
  - `Tony` is not a palindrome
  - `racecar` is a palindrome
- Mark any doubles within the text by placing a 2 between them - **(10 points)**
  - `Mississippi` -> `Mis2sis2sip2pi`

#3 Array Manipulations (40 points)

Working with lists and loops is a critical part of programming. Lists hold data that we often want to query for information or calculate statistics for (e.g., averages, medians). You can complete some or all of the following functions around lists.

- Complete the total of a list of numbers (i.e., add them up!) - **(5 points)**
- Complete the mean (i.e., average) - **(5 points)**
- Count the number of times a value appears in the list - **(5 points)**
- Find the smallest value *in the list* - **(5 points)**
- Find the largest value in the list - **(5 points)**
- Calculate the median of the list (i.e., the middle number if the list contains an odd number of values or the average or the two middle numbers if the list is even) - **(5 points)**
- See if any value within the list is ten times the value of any other value in the list - **(10 points)**