# CECS 424 Assignment 2 - Scripting Languages with Ruby

Due Date: 15 MAR 2019

20 points

**Assignment Description.** The purpose of this assignment is to familiarize you with the features of scripting languages and demonstrate the power that these languages can offer for certain tasks.

**Part One - Coin Arranger.** We're going to play a game with this assignment. We begin with ten coins in a row (five heads and five tails) tightly packed with no gaps between adjacent coins.

**HHHHHTTTTT**

The object is to rearrange the coins into an alternating heads/tails pattern, i.e., one of the following two patterns:

**HTHTHTHTHT**
**THTHTHTHTH**

Again, there are no gaps between adjacent coins. The rearrangement must be accomplished in five "moves" where a move consists of taking any two touching coins and moving them into a two-coin gap, if such a gap exists, or to one end of the row, if no such gap exists. For example, initially there is no two-coin gap, so a possible move would be to take the "HT" coins in the center and move them to the right end, yielding

**HHHH--TTTTHT**

Now there is a two-coin gap, so on the next move we would have to take two adjacent coins and move them into the gap, e.g., we could try

**HHHHTHTTT--T**

As these examples illustrate, the coins are not allowed to switch places during the move, i.e., of the two moved coins, the one on the left before the move must remain on the left after the move. Also, moving coins to the end of the row is illegal if there is a two-coin interior gap. The two coins that are moved must be touching; so for example the isolated "**T**" at the right end in the last picture above cannot be moved on the next move.

Write a Ruby program to play this game. Allow for a maximum of five moves per playthrough. Use strings and display the string after each move. For each move, display the string and let the user select the position of the left of the two characters to move by spacing to it with the cursor under that left character of the two. Then print the string again and let the user space to the left position to where the two characters should be moved. Use the *split* method with an empty string argument to convert the string to an array. Make the move and use the *join* method to convert the array to a string. Repeat until five moves have been made.

**Part Two - Array Partitioner.** Write a Ruby program to partition an array. Read in $n$ values to an array, and a test value $x$. Rearrange the array so that the elements up to and including index $p$ are $<= x$ and the elements from $p+1$ to n are $> x$. Elements may be repeated. The test value, $x$, may be larger than all values or smaller than all values or in between somewhere. You may only visit each element once, and may not copy it to another array. For each test case show the input array, the output array, and the partition index $p$.

For example, given: **28 26 25 11 16 12 24 29 6 10** with test **17** the result might be: **10 6 12 11 16 25 24 29 26 28** with partition index **4**.

An outline of an algorithm is as follows:

1. Start with markers at each end. Move markers towards each other until a wrongly placed pair is encountered. Allow for $x$ being outside the range of array values.

2. While the two markers have not crossed over:
    (a) Exchange the wrongly placed pair and move both markers inward by one.
    (b) Extend the left marker while elements are less than or equal to x.
    (c) Extend the right marker while elements are greater than x.
3. Return the partition index $p$ and the partitioned array.

**Deliverables.** Submit your source code files through Beachboard Dropbox. Make sure that you have adequately commented your original source code, else I will not grade it and you will receive zero points for the assignment!