# Thomas Martin

# CS-300-H2990 DSA: Analysis and Design 23EW2

# 2-3 Assignment Vector Sorting - Code Reflection and Pseudocode

# Southern New Hampshire University

**November 11, 2023**

**Code Reflection**

Sort algorithm, implemented the sorting logic based on the "bid.title" field. The selectionSort method iterates through the array and finds the minimum element in the unsorted portion, then swaps it with the first unsorted element. This process is repeated until the entire array is sorted.

In the main method, invoked selectionSort with a sample array of Bid objects. Also collected and reported timing results by measuring the time taken before and after the sorting operation.

For quicksort, implemented the quickSort method which recursively partitions the array based on the "bid.title" field. The partition method is responsible for placing the pivot element in its correct position and partitioning the array into two halves.

This code reflection summarizes the implementation of selection sort and quicksort, focusing on sorting based on the "bid.title" field. The timing results are collected and reported to evaluate the efficiency of the sorting algorithms.

**Pseudocode**

Implement the quickSort function(bids, begin, end):

Initialize a variable middle to 0.

If begin >= end, return.

Call partition(bids, begin, end) and store the result in middle.

Recursively call quickSort for the lower partition (begin to middle).

Recursively call quickSort for the higher partition (middle + 1 to end).

Implement the selectionSort function(bids):

Initialize a variable smallest to 0.

Initialize a variable largest to the size of bids.

Use a loop with a variable place to iterate from 0 to largest - 1:

Set smallest to place.

Use another loop with a variable j to iterate from place + 1 to largest - 1:

Compare bids[j] title with bids[smallest] title, if smaller, set smallest to j.

If smallest is not equal to place, swap bids[place] and bids[smallest].

Implement the main function:

Process command line arguments to get the CSV file path.

Define a vector to hold all the bids.

Define a clock\_t variable ticks to measure the execution time of sorting algorithms.

Implement a loop until the user chooses to exit:

Display a menu with options to load bids, display all bids, selection sort, quick sort, and exit.

Based on the user's choice, perform the appropriate action:

Load bids using loadBids and measure the time taken.

Display all bids using a loop and the displayBid method.

Sort bids using selectionSort or quickSort and measure the time taken.

Print the number of bids read and the time taken in seconds.

Exit the loop when the user chooses to exit and print "Good bye."