Graciela Orozco

CPSC 335-02 Project #3: 3 Sort Race Complexity Analysis

Overall Complexity

The program runs at O(n^2) + O(n^2) + O(N) complexity due to the three sorting algorithms. The algorithms do not rely on each other which is why they are added to each other instead of multiplied.

insertionSort()

The complexity of insertionSort() is O(n^2). In the worst case, the algorithm would use two loops to loop through each element of the given array. One loop will iterate for the chosen element and compare itself with the other elements in the rest of the while loop based on conditions such as checking for bounds and making sure it is a greater value than its lesser values.

quickSort()

The complexity of quickSort() is also O(n^2) in its worst case. This would indicate that the algorithm selected a bad pivot for the array causing the program to loop throughout the whole array for each pivot not already selected. This algorithm, however, has the potential to be better than others if the correct pivot is chosen. By having a good pivot for the data, a majority of the elements can be sorted faster.

mergeSort()

The complexity of mergeSort() is O(n * log n). The program automatically separates the data in half until they are down to their single element. Afterwards, the algorithm matches the element back up in order until all of the pieces are put back together. This algorithm is fast since it relies on future calls of the program to have already organized the data, allowing the program to check for less and worry about its current iteration.

createRow(), createData(), createTableRow(), pressedAgain(), main()

The createRow(), createData(), createTableRow(), pressedAgain(), and main() functions rely on O(1) time. Though they depend of the sorting algorithms, the functions themselves are constant since they are just basic commands that will use the same amount of time regardless of what is done. The three create functions are used during the algorithm calls and only multiply their time by O(1) since they create HTML elements. The other two refer to organizing the data, but do not involve themselves with any modification to the input..