Programming Project Report

Name: Cameron Eddy
Date: 7/6/2024

Academic Integrity Statement: I pledge that I have neither given nor received unauthorized help on this programming assignment.

Problem Statement:
The objective of this assignment was to implement a dual-pivot quicksort algorithm and compare its performance against the traditional quicksort algorithm. To compare the performance of the two algorithms I tested both with 1000, 100000, and 1000000 samples and put both into an excel file with a graph to compare their respective runtimes.

Design:
To implement the dual-pivot quicksort algorithm, I used the following approach. I selected the low and high as the two pivot points. Then, I partitioned the array into three segments based on the pivot values. After partitioning the array I switched low with the lt-1 and high with gt+1. This is because it would move them to the endpoints of the less-than-segment and greater-than-segment per their respective positions. After that I checked if gt was still gt or equal to lt and recursively called the dual quicksort again. To break our of the function, I had a base at the top that if low was ever greater then or equal to high, then the array was sorted and we should stop

For the traditional quicksort, I used the code from the slides and created a menu based on the labs to compare runtimes. I also had two for loops that would put the unsorted data into one .txt file and another for loop that would put the sorted data into its own .txt file.

Implementation:
To implement this program, I first started with creating the menu and getting the regular quick sort function to work. Then when setting up the dual pivot quick sort function I had to first check to make sure the function would switch the low and high indexes if need be. Then I used hard-coded values will I tested the partitioning of the array. I got the function to work, however, it was slower than the regular quick sort function so after emailing Dr. Streeter and receiving some suggestions to call the recursive function my code got faster.

Testing:
For testing if the sorted function was working I used 10 and 100 as ranges for my two test cases for the function in my script file with 10 elements in each array. This kept the .txt files short but showed that they successfully sorted the data.

Conclusions:
The result of this assignment was a success. Though it was weird when I first got it to work but it ran slowly. After I figured out how to get it to run faster though, the assignment was practically done. It took me about 5-8 hours to complete.