Name: Samyak Piya

CS315 Programming Assignment 1 Writeup

Compile Instructions

To compile the program, unzip the file and open the command prompt. From the command line, change the directory to where the unzipped files are located and type in the command: python3 main.py

Estimate and Graph Runtime

The following table presents the runtime of each sorting algorithm on the 9 different data sets provided. This was calculated by putting variables into the sorting functions with an initial value of 0 that increments each time a comparison is made to an element from the list being sorted.

	A	B	C	D
1		RUNTIME		
2	Arrays	Insertion Sort	Merge Sort	Quick Sort
3	pokemonSortedSmall	165	1238	13695
4	pokemonReverseSortedSmall	13672	1238	9523
5	pokemonRandomSmall	6934	1238	1203
6				
7	pokemonSortedMedium	431	3808	93096
8	pokemonReverseSortedMedium	92330	3808	49429
9	pokemonRandomMedium	46589	3808	4226
10				
11	pokemonSortedLarge	799	7776	319600
12	pokemonReverseSortedLarge	316730	7776	157546
13	pokemonRandomLarge	163108	7776	12164

Insertion Sort has a time complexity of O(n) in the best case, and $O(n^2)$ in the worst case. Looking at the values in column B of the table, we can conclude that this behavior is normal as elements are sorted very quickly when a mostly sorted array is provided. However, when the array is sorted in reverse to what we want, it takes the longest time amongst any other sorting algorithms.

Merge Sort has a time complexity of O(n*log(n)) in all its cases (best, average, worst). Looking at the values in column C of the table, we can conclude that this behavior is normal as for a given 'n' number of values, the number of comparisons made always evaluates to the same number

Quick Sort also has a time complexity of O(n*log(n)) in the best case, and O(n^2) in the worst case. Looking at the values in column C of the table, we can conclude that this behavior is normal as when a mostly sorted array is provided, quicksort has to make a greater number of comparisons and is only faster than merge sort in the case where the number of elements 'n' is very small, and the elements of the array are randomly permutated.

Some important details about my implementation:

1. The 'global' keyword is used within the sorting functions (especially needed for recursively defined merge-sort and quick-sort functions) so that the value being tracked is not lost in subsequent recurrence calls.
2. The math library is imported to utilize the math.floor() function.
3. Some helper functions print_array(array) and print_runtime(filename, sort_algorithm, run_time) are created to print arrays and the runtime evaluated for sorting each dataset respectively.
4. When merge_sort(array, p, r) and quick_sort(array, p, r) is called, the length of the array – 1 is passed as the third parameter (r).