Design and Analysis of Algorithms (CS253)
Practical Assignment

Team Members: Ankit Dash(211CS212), Meet Banthia(211CS232)

Question 1

- The input files we use for testing are generated dynamically during run time.
- The 3 pivot choices have been implemented in the quick_sort.py file.

Question 2

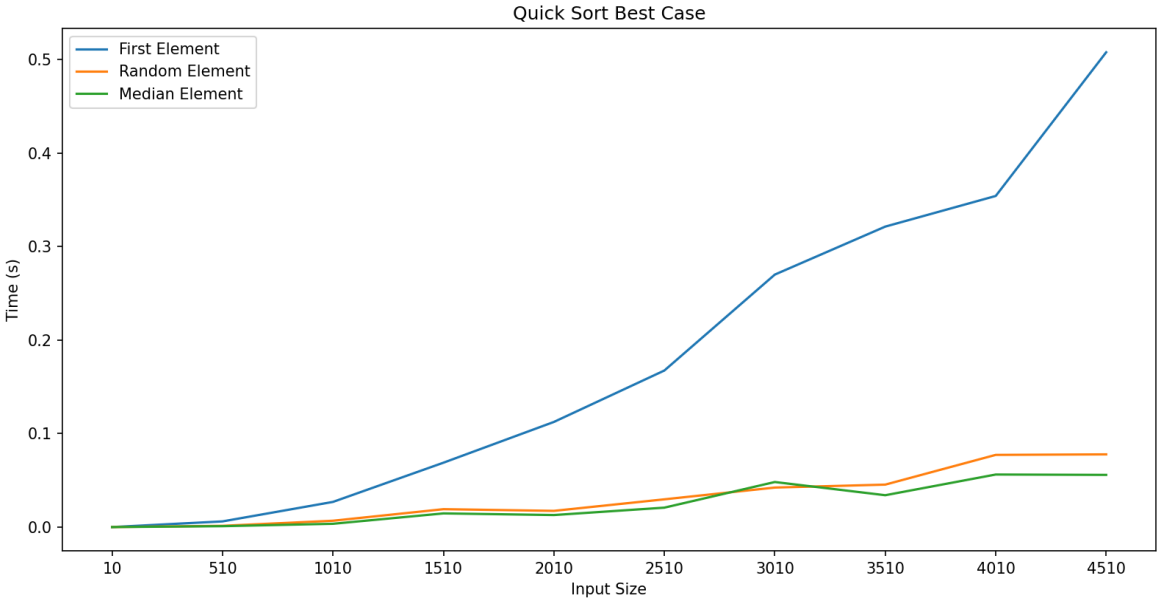
- Device Specifications:
  HP-Pavilion-14: 11th Gen Intel(R) i5, 2.5 GHz processor, and an x64 bit Windows OS having 8GB of RAM.
- We used Python's built-in time module and for our specific purpose, we used time.perf_counter().
- The experiment was run around 12 times with different array sizes to obtain stable measurements which we show in the snapshots below.
- The times reported have been stored in each sort's csv_files folder.
- Three types of inputs were selected:
  - ➢ An increasing array.
  - ➢ A randomly generated array.
  - ➢ A decreasing array.

  The array's lengths selected were from 1 to 1000 with a step size of 10 for fastening the process.
- Yes, once a particular type of array was generated, the same array was used as input for all the sorting algorithms to avoid bias and give reliable results.

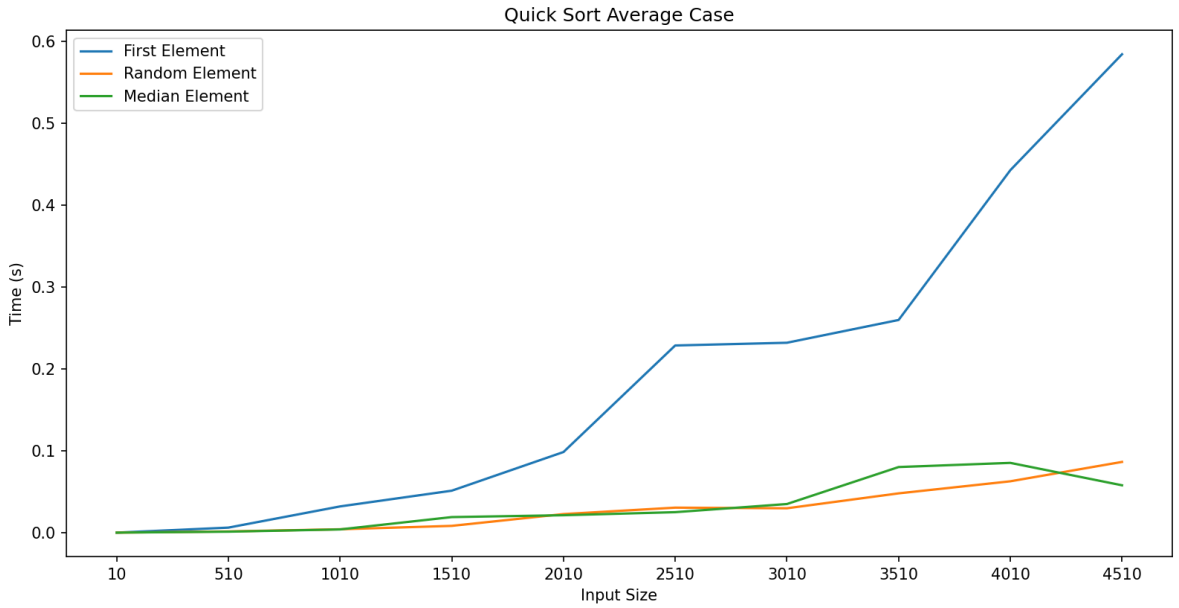
Question 3

We simulated various cases of quick sort for each of the 3 pivot choices and took the relevant results for plotting the best case, average case, and worst cases of quick sort.

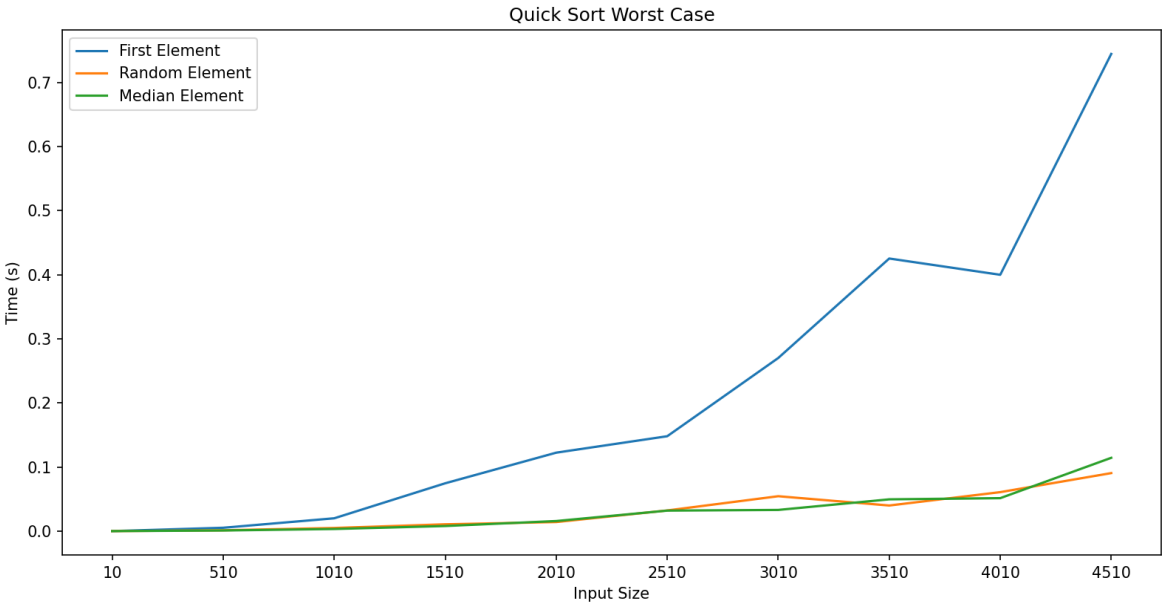
- Graph the best case running time as a function of input size n for all three versions (use the best case input you determined in each case in part 1).



- Graph the average case running time as a function of input size n for all three versions.



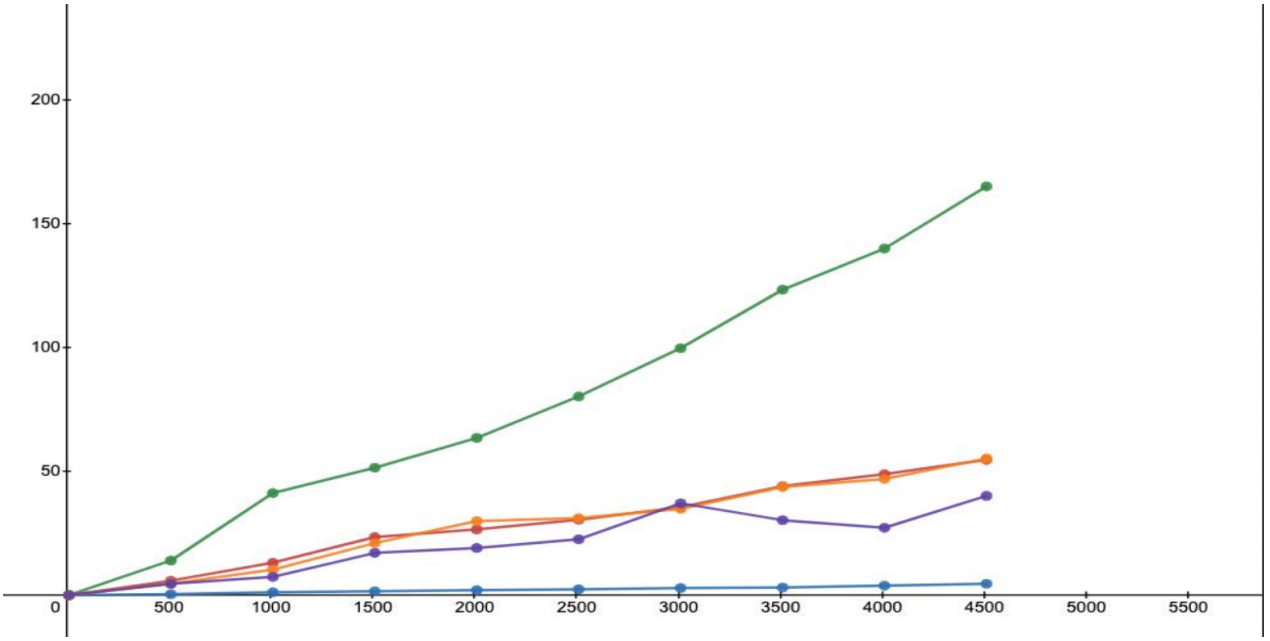
- Graph the worst-case running time as a function of input size n for all three versions (use the worst-case input you determined in each case in part 1).



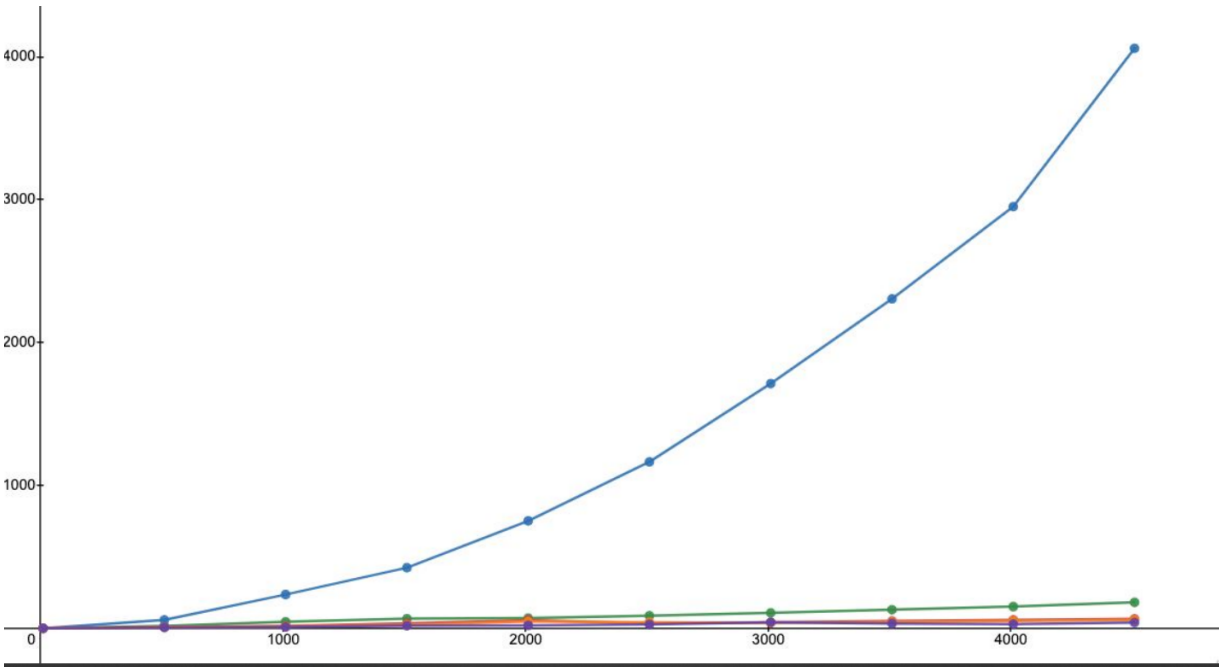
Question 4

Which of the five sorts seems to perform the best (consider the best version of Quicksort)?
Ans:
Color Schema:
Green: Heap Sort
Blue: Insertion Sort
Red: Merge Sort
Orange: Quick Sort
Purple: Radix Sort

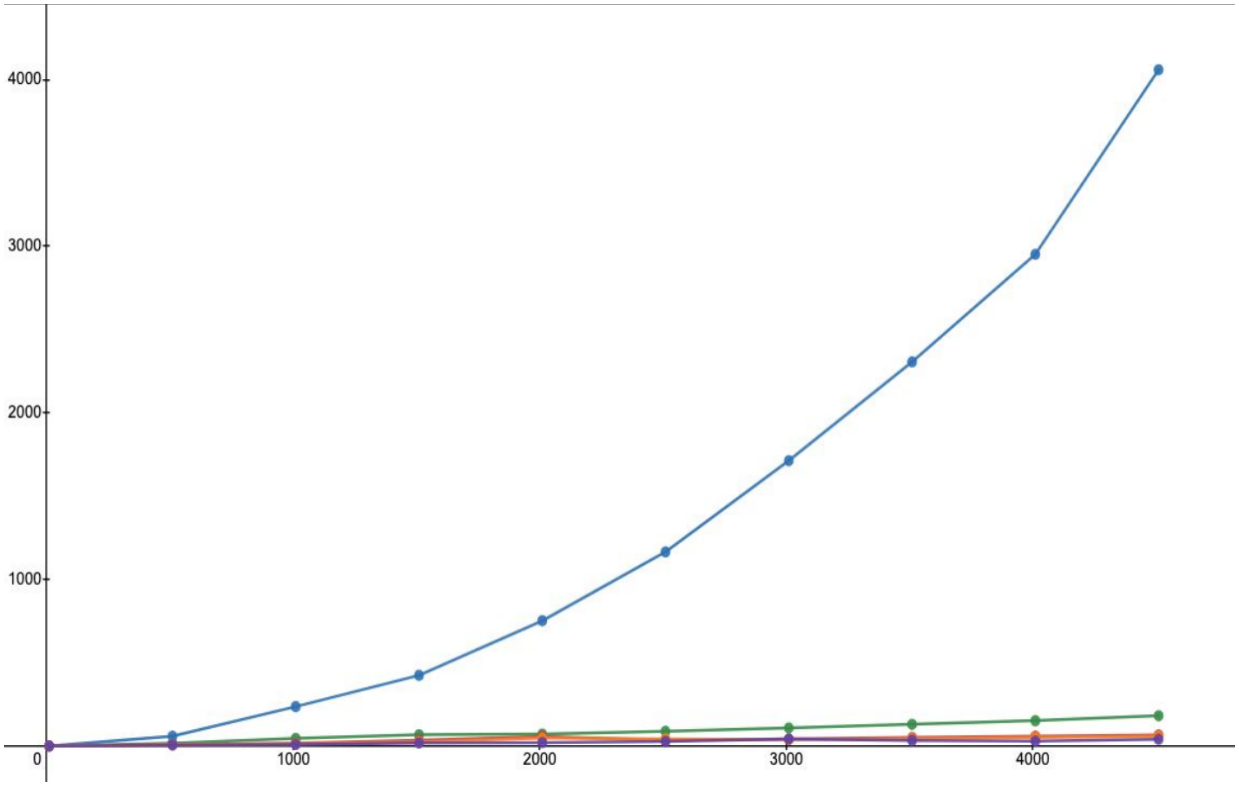
- Graph the best-case running time as a function of input size n for the five sorts.



- Graph the average case running time as a function of input size n for the five sorts.



- Graph the worst-case running time as a function of input size n for the five sorts



Question 5

Analyze your data to see if the number of comparisons is correlated with execution time. Plot (time / #comparisons) vs. n and refer to these plots in your answer.

Ans: Yes, the time or number of comparisons does correlate with the number of inputs. It is because the number of comparisons increase the time it takes to sort the array.The plots mentioned above demonstrate the fact that as the number of inputs increases the time taken to sort the array also increases with varying degrees of proportion.