# Name: Harsh Kaushik

# Entry no:2023AIB1006

# Department: Artificial intelligence and Data engineering

# Course ID:CS205

# System specification:

1. RAM: 16GB
2. Graphics card: RTX4060 8GB VRAM
3. CPU: Ryzen 7 7840HS
4. OS: Windows 11

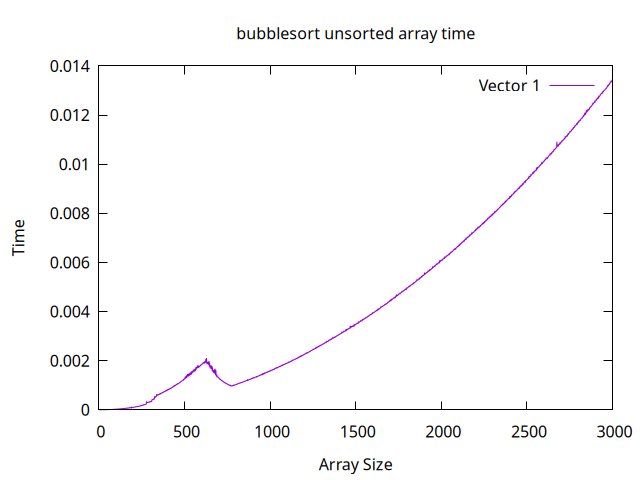
SORTING ALGORITHMS

* Bubble Sort
* Insertion Sort
* Selection Sort
* Merge Sort
* Quick Sort

# Bubble sort

* Bubble sort is one of the most basic sorting algorithms which involves multiple iterations of array and swapping the neighbouring elements when the condition is met.
* Bubble sort is a Stable algorithm.
* Best Case: omega(n^2) (since we haven’t put any break statement inside algorithm, it keeps on iterating), happens in sorted array.
* Worst Case: O(n^2), happens in reverse sorted array.
* A modification can be done to bubble sort algorithm by adding break statement at the end of n iterations, which can lead its best case to have time complexity of omega(n), in case of already sorted array.

# Time taken in case of unsorted array



* Here, we got an observation that from about 600 to 800 array lengths, the time was decreasing, which is highly unusual, the reason for this might very well be because of the compiler optimization or cache effect which could lead to faster access time.
* Apart from 600 to 800 area, we get a curve proportional to n^2 which is what we expect from Bubble Sort.

# Time taken in case of sorted array

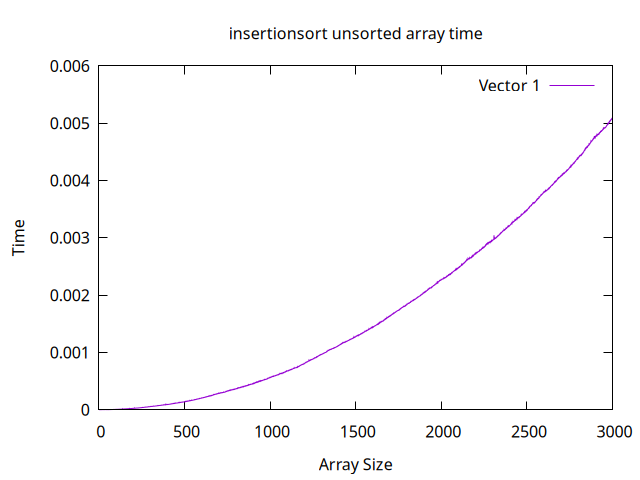
# 

* Got curve proportional to n^2 as expected.

# Insertion Sort

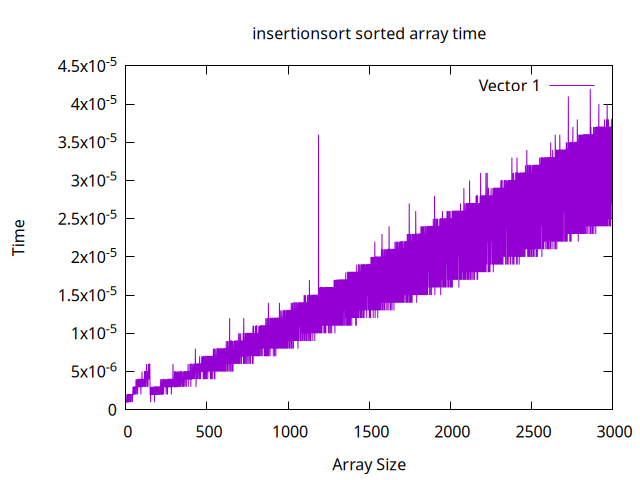
* Insertion sort is based on the principle of keeping sorted array to the right and increasing its length by selecting the next element to the sorted array and putting it in the right position inside the sorted array via swapping.
* Insertion sort is a Stable algorithm.
* Best Case: omega(n) (Since there is a break statement inside the Algorithm, if the next element is already bigger than the last element of sorted array => breaks and jumps to the next iteration), happens in case of sorted array.
* Worst Case: O(n^2), in case of reverse sorted array

# Time taken in an unsorted array



* Got curve propotional to n^2 as expected

# Time taken in a sorted array

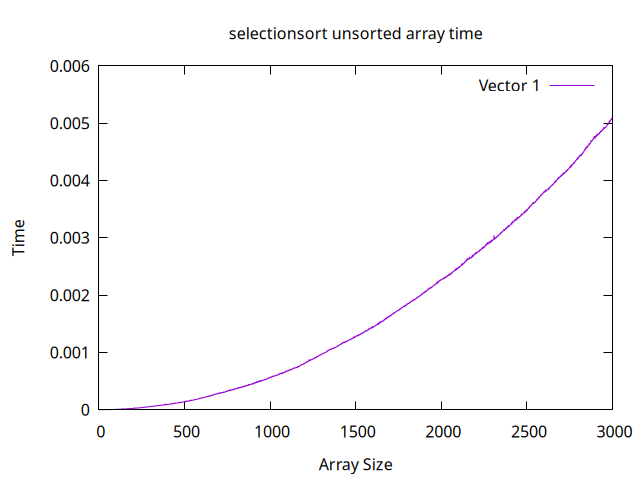


* The line of best fit from the curve is proportional to n as expected (since, sorted array is the best case for insertion sort)

# Selection Sort

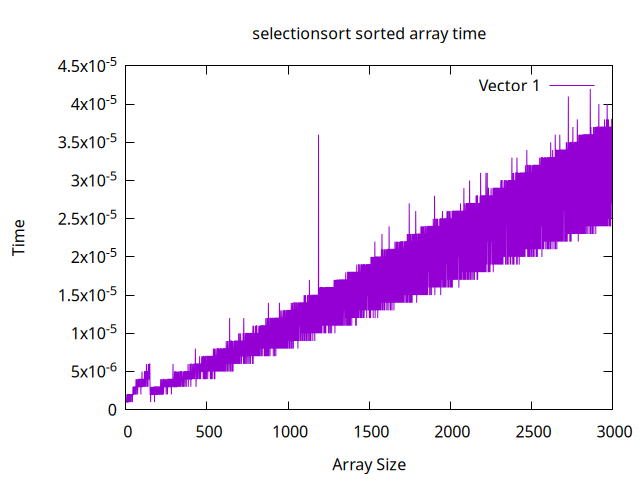
* Selection Sort is based on the principle that there is a sorted array on the left and we keep increasing its length by adding the min element from the rest of the part of the array to the front of the sorted array.
* Selection Sort is an unstable algorithm.
* Best case: omega(n^2) (no break statements to stop iterations), in case of sorted array.
* Worst case :O(n^2) (technically every case is best case and worst case both), in case of reverse sorted array.

# Time taken in unsorted array



* Curve proportional to n^2 as expected.

# Time taken in sorted array

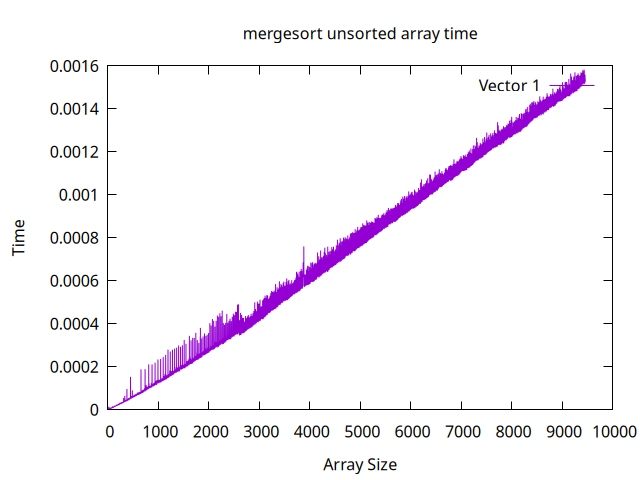


* Due to variation in times, the curve is bit unclear but it should be proportional to n^2.

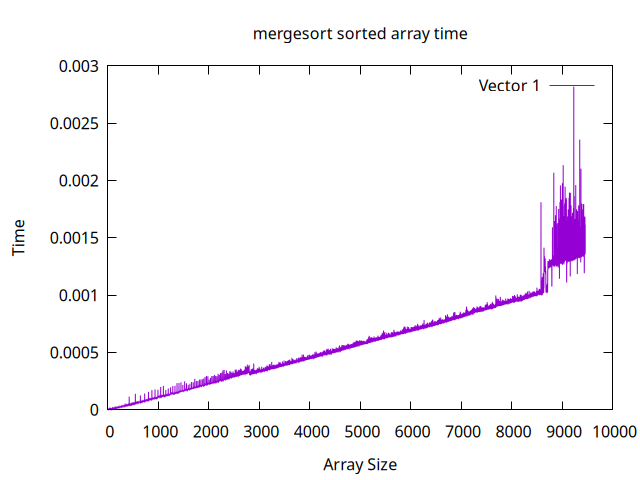
# (4) Merge Sort

* Merge sort is based on the divide and conquer strategy by first dividing the array recursively, sorting it and then recursively joining it.
* Merge sort is a stable algorithm.
* Best case: omega(n.log(n)), in case of already sorted array.
* Worst case: O(n.log(n)), worst case doesn’t depend upon initial arrangements of elements.

# (A)Time taken in unsorted array



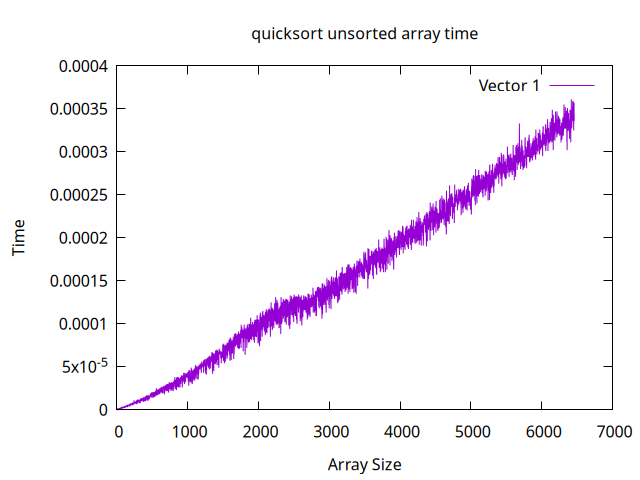
# (B)Time taken in sorted array



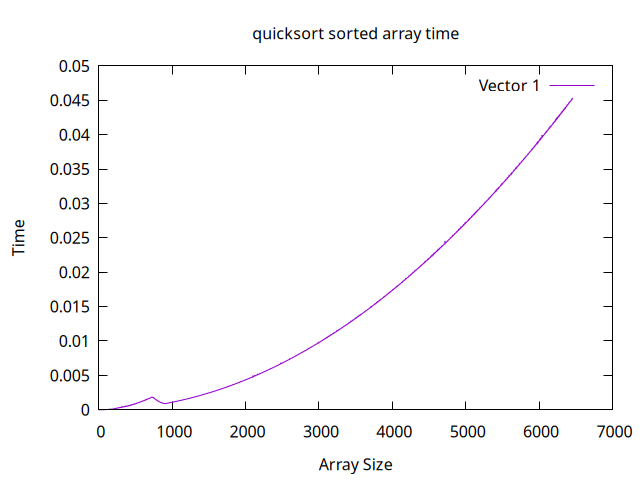
# 5. Quick Sort

* Quick sort is based on divide and conquer strategy in which we select a pivot which divides array into 2 parts (all smaller elements on left and all larger on the right) and keep doing this recursively.
* Best case: omega(n.log(n)), happens when the pivot chosen is median of the array (we can find approximate median (not exact)-which divides the array in about 70-30 ratio using median of medians algorithm)
* Worst case: O(n^2), in case when we select a pivot which is either the largest or smallest element, in our case of selecting the last element as pivot, sorted and reverse sorted array are the worst cases.

# (A)Time taken in unsorted array



# (B)Time taken in sorted array



* Since sorted array is the worst case for quicksort(if pivot is last element), it is giving a curve proportional to n^2 which is indeed expected.