NOTE:PLEASE RUN MY PROGRAM THROUGH IDE’S LIKE ECPIPSE OR EXLIPSE , BECAUSE EVEN THOUGH MY PROGRAM WORKS 100% WELL ON MY COMPUTER,IT DOES NOT RUN ON THE ITU SERVER.YOU CAN ALSO RUN THE PROGRAM ON NORMAL LINUX TERMINAL OR THROUGH AN IDE AS I MENTIONED.THANK YOU ☺

REPORT

Q1)

|  |  |  |  |
| --- | --- | --- | --- |
|  | hs-set-10k.txt | hs-set-100k.txt | hs-set-1M.txt |
| INSERTION SORT(FULL) | 21.2 SECONDS | 2000 SECONDS | 3.5 hrs |
| INSERTION SORT(FILTER) | 6.9 SECONDS | 695 SECONDS | 70mins |
| MERGE SORT(FULL) | 0.081 SECONDS | 0.83 SECONDS | 10.0 SECONDS |
| MERGE SORT(FILTER) | 0.051 SECONDS | 0.66 SECONDS | 8.4 SECONDS |

Q2) I realized that when my number of items is low(like 20 items), insertion sort was actually fast, but as my input started growing, merge sort started getting faster. So the reason is that, the complexity of Merge sort is O(n\*log(n)), so merge sort takes the advantage of the complexity and performs better for big inputs. Lastly I can conclude that Merge sort is better than insertion sort for big inputs .

Q3) Sorting through rarity and set is similar to type because all three of them are shot strings , so speed of the program might not change, but unlike sorting by name which has much longer strings, a program may actually takes a bit more time time.

Q4) A stable sorting is a type of sorting that keeps the order of items that have equal key. Both insertion and merge sort are stable algorithms, what could have gone wrong say quick sort which is an unstable algorithm was used instead of insertion sort, is the order within a block of sorted items items might be changing every time we run our code.