CS 325

Guangyu Zhang

3.

a). collect running time

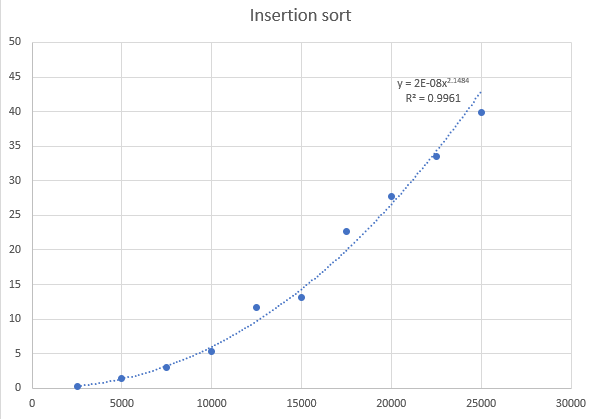
I generate the random arrays by using random function in Python

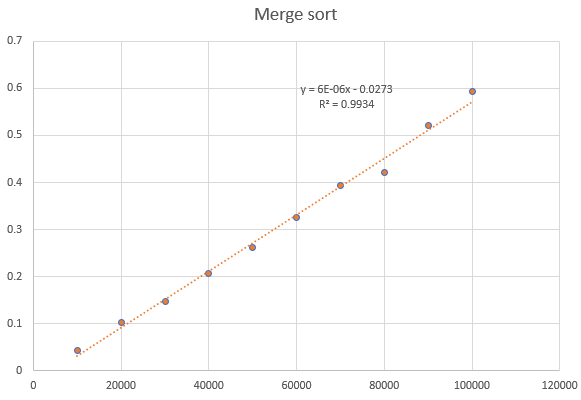
|  |  |
| --- | --- |
| Number(n) | Insertion sort |
| 2500 | 0.31410765647888184 sec |
| 5000 | 1.3504152297973633 sec |
| 7500 | 2.9880361557006836 sec |
| 10000 | 5.380636692047119 sec |
| 12500 | 11.743558645248413 sec |
| 15000 | 13.124891996383667 sec |
| 17500 | 22.597503900527954 sec |
| 20000 | 27.65602707862854 sec |
| 22500 | 33.48346185684204 sec |
| 25000 | 39.86635708808899 sec |

|  |  |
| --- | --- |
| Number(n) | Merge sort |
| 10000 | 0.04283332824707031 sec |
| 20000 | 0.10372638702392578 sec |
| 30000 | 0.14660954475402832 sec |
| 40000 | 0.20744538307189941 sec |
| 50000 | 0.2613043785095215 sec |
| 60000 | 0.3251631259918213 sec |
| 70000 | 0.39394640922546387 sec |
| 80000 | 0.4208981990814209 sec |
| 90000 | 0.5206441879272461 sec |
| 100000 | 0.5934135913848877 sec |

b). plot data and fit a curve

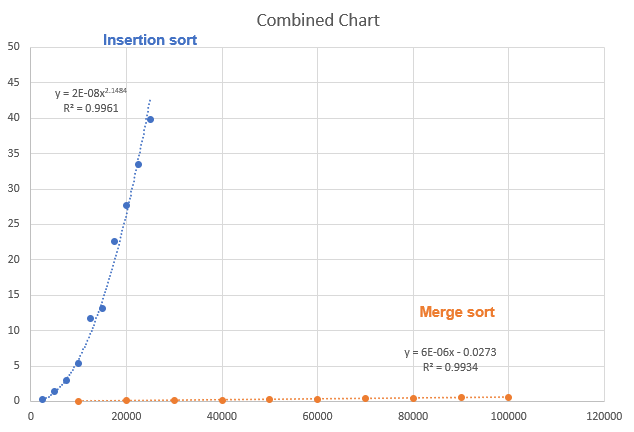
For insertion sort, the best fits curve is polynomial. For merge sort, the best fits curve is linear.





c). combine

merge sort has a lower time consumption than insertion sort



d). prediction

insertion sort:

merge sort:

when x = 200000,

Time of insert sort will approximately be 4895 sec, and time of merge sort will approximately be 1.1727 sec.

e). comparison

My results shows that insertion sort has polynomial complexity, and merge sort has linear complexity. Similarly, the theoretical of insertion sort running time is O(n2) which fits my results. And, the theoretical of merge sort running time is O(nlogn), in my results, the time of halving might be very short which is the logn part. Another possible reason is that the excel cannot draw a graph based on logn function. Therefore, I got the results above.