lab04-exercises

Author’s name: Cuiting Huang

Partner: Hongyang Ye, Siqi Chen

Author affiliation: Northeastern University

City & country location: Zhengzhou, China

E-mail address: huang.cui@northeastern.edu

Abstract – This is the written exercises of lab04 sorting

Index Terms – Sorting, Bubble, Selection, Big O

Question1

Explain what you think the worst-case, big-Oh complexity and the best-case, big-Oh complexity of bubble sort is. Why do you think that?

ANS

Worst-case: We have a reversed array that every single element needs to be sorted. And the big-Oh complexity of this case is O(n^2).

For the worst-case we need to set every step costs a constant time, we need n steps to sort a single element from the beginning to the end. For n elements in an array, in total we need n*n steps.

Best-case: We have a sequential array that all the elements are already sorted. And the big-Oh complexity of this case is O(n).

For every elements have already been sorted, we can just walk through all the elements for a time.

i.e., we run the internal for loop for one time, which takes n steps. If we found all the elements are sorted, we break the loop.

Question2

Explain what you think the worst-case, big-Oh complexity and the best-case, big-Oh complexity of selection sort is. Why do you think that?

ANS

For selection sort, we could only know which element is the smallest after the iteration of the end of the unsorted portion. Even if the array is sequential sorted ,we must also iterate until the end.

For one element, we conduct function find Minimum for one time, which takes n steps.

For every elements, it needs n*n steps.

There is no difference between every case of the selection sort. The big-Oh complexity of selection sort is always O(n^2).

Question3

Does selection sort require any additional storage (i.e. did you have to allocate any extra memory to perform the sort?) beyond the original array?

ANS

No. According to our code, we do not allocate any extra memory to perform the sort.

Question4

Would the big-Oh complexity of any of these algorithms change if we used a linked list instead of an array?

ANS

The big-Oh complexity remains the same for a linked list.

For bubble sort, since it exchanges the locations of two adjacent elements. It works the same for two adjacent nodes in a linked list.

For selection sort, we can use two pointers to change the location of the selected element and the minimum. Therefore, the big-Oh complexity also remains the same.

Question5

Explain what you think big-Oh complexity of sorting algorithm that is built into the c libraries is. Why do you think that?

ANS

I think the big-Oh complexity of sorting algorithm that is built into the c libraries is O(nlogn).

According to the runtime of these sorts we got in the google sheet. The csort always runs so faster than the others. For the best case of bubble sort,the big-Oh complexity of this case is O(n). I guess the sort built in C-library should be more quicker, and O(nlogn) is a good choice.