ShanghaiTech University

CS101 Algorithms and Data Structures

Fall 2021

Homework 3

Due date: 23:59, October 17, 2021

1. Please write your solutions in English.
2. Submit your solutions to gradescope.com.
3. Set your FULL NAME to your Chinese name and your STUDENT ID correctly in Account Settings.
4. If you want to submit a handwritten version, scan it clearly. Camscanner is recommended.
5. When submitting, match your solutions to the according problem numbers correctly.
6. No late submission will be accepted.
7. Violations to any of the above may result in zero grade.

1: (2'+2') Sorting Practice

Given array:

$$1, 5, 3, 8, 7, 2, 4, 6$$

we want to sort this array in ascending order in-place.
Please **show your steps** taken by each type of sorting method on this array:

Question 1. ***Quicksort.*** *After each partition during the algorithm, write the ordering of the list,* ***circle*** *the pivot that was used for that partition, and* ***underline*** *the sub-array being partitioned. Assume that the pivot is always the first item in the sublist being sorted.*

Example: $\underline{[1]\ 5\ 3\ 8\ 7\ 2\ 4\ 6}$
1 $\underline{[5]\ 3\ 8\ 7\ 2\ 4\ 6}$
1 $\underline{[4]\ 3\ 2}$ 5 $\underline{[7]\ 8\ 6}$
1 $\underline{[2]}$ 3 $\underline{4\ [6]}$ 7 $\underline{[8]}$
1 2 $\underline{[3]}$ 4 5 6 7 8

Question 2. ***Merge sort****. Show the intermediate merging steps. In each merging step, circle the subarray that is to be merged.*

[1] [5] [3] [8] [7] [2] [4] [6]
[1 5] [3 8] [2 7] [4 6]
[1 3 5 8] [2 4 6 7]
1 2 3 4 5 6 7 8

2: (4×1') Identifying Sorts

Below you will find intermediate steps in performing various sorting algorithms that sort the input in ascending order on the same input list. The steps **do not necessarily represent consecutive steps in the algorithm** (that is, many steps are missing), but they are in the correct order. For each of them, **select the correct algorithm from the following choices**: insertion sort, bubble sort, quicksort (first element of sequence as pivot), and mergesort.

Input list: 5, 6, 9, 4, 1, 2, 7, 10, 8, 0, 3

Question 3.
5, 6, 9, 1, 4, 2, 7, 10, 8, 0, 3
5, 6, 1, 4, 9, 2, 7, 10, 0, 3, 8
1, 4, 5, 6, 9, 0, 2, 3, 7, 8, 10
mergesort

Question 4.
4, 1, 2, 0, 3, 5, 6, 9, 7, 10, 8
1, 2, 0, 3, 4, 5, 6, 9, 7, 10, 8
0, 1, 2, 3, 4, 5, 6, 7, 10, 8, 9
quicksort

Question 5.
4, 5, 6, 9, 1, 2, 7, 10, 8, 0, 3
1, 4, 5, 6, 9, 2, 7, 10, 8, 0, 3
1, 2, 4, 5, 6, 9, 7, 10, 8, 0, 3
insertion sort

Question 6.
5, 6, 4, 9, 1, 2, 7, 10, 8, 0, 3
5, 6, 4, 1, 2, 7, 9, 8, 0, 3, 10
5, 6, 1, 2, 4, 7, 9, 8, 0, 3, 10
bubble sort

3: (3×1') Single Choice

The following questions are single choice questions, each question has **only one** correct answer. Select the correct answer.
Note: You should write those answers in the box below.

Question 7	Question 8	Question 9
A	C	C

Question 7. *Consider the quicksort algorithm which sorts elements in ascending order using the first element as pivot. Then which of the following input sequence will require a maximum number of comparisons when this algorithm is applied on it?*

(A) 22 25 56 67 89

(B) 52 25 76 67 89

(C) 22 52 67 25 76

(D) 52 25 89 67 76

Question 8. *Which of the following statements is **NOT** true?*

(A) The worst case time complexity of quicksort is $O(n^2)$.

(B) Given 2 sorted lists of size m and n respectively, and we want to merge them to one sorted list by mergesort. Then in the worst case, we need $m + n - 1$ comparisons.

(C) The time complexity of quicksort, compared with mergesort, is less affected by the initial order of the input array.

(D) Comparing to quicksort, mergesort requires additional space complexity.

Question 9. *Given extra information about the input array, we may design sorting algorithms that perform faster than $O(NlogN)$. Which of the following prior knowledge will lead to worst time complexity **slower** than $O(N)$?*

(A) Knowing the input array has no more than N inversions.

(B) Knowing the input array has exactly $(N^2 - N)/2$ inversions.

(C) Knowing the input array has less than N pairs of numbers that are not inversions.

(D) None of the above.

Question 10. ***(4')Stable sort***

We say a sort is "stable" if there is an implementation that it ***always*** *preserves the original order of equal elements in an array without using additional space. For example, if we have an array:* $4, 2, 3_a, 3_b, 1$*, after sorting it has a chance to become* $1, 2, 3_b, 3_a, 4$ *(* 3_a *and* 3_b *are equal when comparing), then this sorting algorithm is* ***NOT*** *stable.*

Among insertion sort, quicksort and mergesort, which of them are stable? (1') Briefly explain your answer. (3')

Solution

Insertion sort and mergesort are stable.

As for insertion sort. If we define that every number should be inserted before the last number that is strictly greater than itself when probing from back to front, then any pair of numbers that equal would be inserted in their original sequence, and be in the same sequence after their insertion respectively, since none of them can be greater than the other.

As for mergesort. In a mergesort we divide the whole array into pieces. When it comes to merging two subarrays, if we always take the number from the first subarray first when equal number from both subarray are under considering, then we can keep the original sequence of the two equal numbers.

Quicksort is not stabe, because its uncertainty in choosing a pivot, and the swap process after each traverse.