Ch. 7.1-7.4 Sorting 習題

Q1

- One day, one of your friends challenges you to play a game which requires you to guess a number, within the rage of 1 to 100, that he has set in mind. If you get a wrong guess, he would tell you the value is higher or lower than your guess. What is the maximum number of guesses that you can guarantee to get the right answer?

Ans 1:

- The question is equivalent to find the existence of a element in sorted number array. If the guess is higher than the number, the player return “Too high”. If the guess is lower, then return “Too low”. As the binary search, each guess will exclude half of all possible values. Thus, the time complexity is O(log n).
- The maximum number of guesses is $\log_2 100$ -> 7 times

For example: the correct number is 33.

1, 2, ..., 33,...,50, ...,99, 100 ->Choose 50, too high

1, 2, ...,25 , 33,...,49 -> Choose 25, too low

26, ...,33, ... 37, ...49 -> Choose 37, too high

26, ... ,31, 32,33, ... 36 -> Choose 31, too low

32,33, 34 ,35, 36 -> Choose 34, too high

32,33 -> choose 32, too low

33 -> -> choose 33, exactly!!!

Q2

- Please define what is a stable sorting algorithm and an unstable sorting algorithm, and then give an example to each type of sorting algorithm.

Ans 2:

1. Stable sorting : The order of two equal keys ends up in the same order after sorting.
   - Examples: Insertion sort, Merge Sort, Bubble Sort.
2. Unstable sorting: The order of two equal keys may end up in different orders after sorting
   - Examples: Heap Sort, Quick Sort.

Illustration:

- Original: 4, 3, 5, 5, 7, 10, 9
- After stable sorting: 3, 4, 5, 5, 7, 9, 10
- After unstable sorting: 3, 4, 5, 5, 7, 9, 10

Q3

- (a) Apply step-by-step insertion sort to the following list, [5, 6, 3, 8, 2].
- (b) Apply step-by-step merge sort to the following list, [50, 10, 90, 30, 70, 40, 80, 60, 20]

Ans: (a) insertion sort

- 5, 6, 3, 8, 2
- -> 5, 6, 3, 8, 2
- -> 5, 6, 3, 8, 2
- -> 3, 5, 6, 8, 2
- -> 3, 5, 6, 8, 2
- -> 2, 3, 5, 6, 8

Ans: (b) merge sort

- 50, 10, 90, 30, 70, 40, 80, 60, 20
- 10, 50, 30, 90, 40, 70, 60, 80, 20
- 10, 30, 50, 90, 40, 60, 70, 80, 20
- 10, 30, 40, 50, 60, 70, 80, 90, 20
- 10, 20, 30, 40, 50, 60, 70, 80, 90

Q4

- Suppose that we are sorting an array of eight integers using Quick Sort, and we have just finished the first partitioning with the following resulted array: 2 5 1 7 9 12 11 10. What could be the pivot for the first partition?

a) The pivot could be either 7 or 9.

b) The pivot could be 7, but not 9.

c) The pivot could be 9, but not 7.

d) Neither 7 nor 9 is the pivot.

Ans 4:

- All elements in the left (of the pivot) should be smaller than the pivot.
- All elements in the right (of the pivot) should be larger than the pivot.

Ans: (a) The pivot could be either 7 or 9.

- If 7 is pivot, 2 5 1 **7** 9 12 11 10: Correct!
- If 9 is pivot, 2 5 1 7 **9** 12 11 10: Correct!

Q5

- **The best time complexity of sorting algorithm based on comparison and swap is** $O(n \log n)$**.**

Ans 5:

- The sorting result can consider as a binary decision tree if only compare() and swap() function can be used.
- Explanation:
  1. To sort an array of n elements, the possible leaf node in the decision tree is **n!**. Since the decision tree is a binary tree, the smallest height must be $\log_2 n! + 1$.
  2. The average path length from the root to a leaf is height – 1 = $\log_2 n!$.
  3. Since $n! = n(n-1)(n-2)\ldots(3)(2)(1) \geq (n/2)^{n/2}$, $\log_2 n! \geq (n/2)\log_2(n/2) = \Omega(n\log n)$