

Problem 1(4×2pts): True or False: For each statement, write “T” if this statement is correct; write “F” otherwise. Please **write your answers in the box below**.

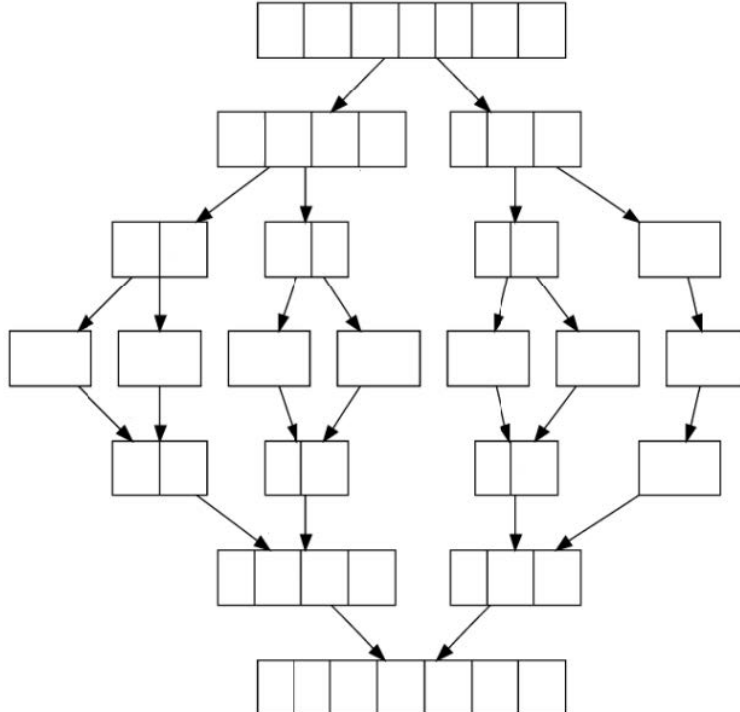
Statement (1)	Statement (2)	Statement (3)	Statement (4)

- (1) Merge sort requires $O(n)$ space complexity.
- (2) In quicksort (sort in ascending order), if we randomly select the pivot, after the first partition operation, the smallest element of the array can be **anywhere but the last position**.
- (3) Quicksort algorithm will have $O(n^2)$ time complexity in the worst case.
- (4) By applying the partition step of quicksort on an **unsorted** array repeatedly, we can get the $k - th$ biggest number of that array with an **average** time complexity of $O(\log(n))$. (k is an arbitrary number)

Problem 2(5pts):

Consider this array: 7, 6, 4, 2, 5, 3, 1.

- (1)(3.5pts) Use **mergesort** to sort this array in ascending order. Show your process in the following figure.



- (2)(1.5pts) How many inversions are there in the array?

Problem 3(3×1pts):

Tom wants to sort his favorite colors in ascending order using quicksort. The original array is:

red, cyan, yellow, gray, green, black, blue, white

After the first partitioning step, it becomes: (“red” is chosen as pivot)

cyan, green, gray, white, black, red, yellow, blue

Known that **NO** elements are equal, we can infer that: (Fill the blanks with “>”, “<”, or “?” if given information is insufficient to judge)

(a) red ----- white (b) yellow ----- gray (c) green ----- black

Problem 4(4pts): Prove that: The time complexity for mergesort is $O(n\log(n))$.