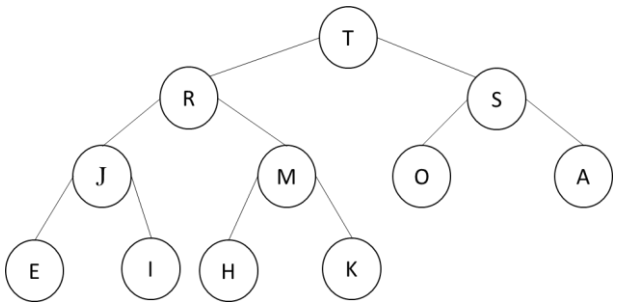
Problem 0. Information (1 point)

Write your name and KAIST ID at the top of **EACH** page on your answer paper.

Problem 1. (10 points)

Consider the following max-heap.



Draw a resulting heap after deleting the maximum key from the heap shown above. Show ALL the intermediate steps. (10 points)

- Note: Please draw a resulting max heap. **Do not give an array-representation.**

Problem 2. (19 points)

Suppose that keys are 3-digit numbers. d_0 is the digit in the rightmost position. For example, the 3-digit number 123 has the following digits: $d_2 = 1$, $d_1 = 2$, $d_0 = 3$. You are given three hash functions as follows. M is a hash table size:

$$h_1(x) = (d_2 + d_1 + d_0) \% M$$

$$h_2(x) = (d_2 + d_1) \% M$$

$$h_3(x) = d_2 \% M$$

Suppose that the following 6 input keys below are inserted into a hash table of size 10 in that order, using one of the hash functions above. Assume that collisions are handled by linear probing, and we do not resize the hash table.

610, 160, 254, 345, 449, 532

1) Rank the three hash functions from best (which causes the least collisions) to worst (which causes the most collisions) for the given 6 input keys and a hash table of size 10.
   Also, explain why you rank functions in that order, based on all the hash values for the given 6 input keys when you use each of the three hash functions above. (worth 14 points)
   - Best:_____ 2nd best:_____ Worst: _____
   - Explain why:
2) If each of the three hash functions above is used on the 6 input keys on a hash table of size 10, what is the load factor? Explain how you compute the load factor. (worth 5 points)
   - h_1:_____ h_2: _____ h_3:_____
   - Explain how you compute the load factor:

Problem 3. (35 points)

Select a critical reason (among A, B, C, D, E) why we use the first one instead of the second one in each statement and provide explanations that justify your selection.

- Note: For each sub-problem, select only one most critical reason and explain why.

A. Save memory space.
B. Stability is required.
C. Avoid re-computation and a potential exponential blow-up in the running time.
D. Guarantee correctness
E. No particular reason.

1) We use quicksort (with random shuffle) instead of mergesort to sort an array of primitive types. (10 points)
2) When we double the size of the underlying array, we rehash all of the keys into new chains in a separate chaining hash table instead of keeping the keys in its old positions. Suppose that hash functions that we use distribute the keys uniformly and independently. (15 points)
3) When computing the 60^{th} Fibonacci number, we use dynamic programming instead of direct implementation of the Fibonacci formula using recursion, as shown below. (10 points)

```
public static long F(int n) {
        if(n ==0) return 0;
        if(n ==1) return 1;
        return F(n-1) + F(n-2);
        }
```

Problem 4. (15 points)

You are given an array of 10,000 non-negative distinct integers from 0 to 9,999 in random order. You are required to rearrange the array such that the integers less than 1,000 are grouped to the beginning, followed by the rest of integers. You are not required to sort all those integers in ascending order. Design an algorithm that performs the task in linear time, using only constant extra memory.

- Note: Adapt an algorithm from what we studied in class.

1) Specify which algorithm you adapt to perform the task described above. (5 points)
2) Explain an algorithm that performs the task above. (10 points)

Problem 5. (20 points)

A string is a sequence of characters. Given two strings X and Y, you are required to design an efficient algorithm that determines whether the string Y is formed by rearranging the characters of the word X. Suppose that the input string consists of at least two alphanumeric characters and its length can be up to 2^{31}-1.

For example, for each of the following pairs, your algorithm should return true:

- dear, read

- beak, bake
- CS206, 206CS
- dusty, study

Otherwise, your algorithm should return false.

1) Design an efficient algorithm that performs the task described above. (worth 15 points)
   - **Notes:** Use the algorithm, ADT, or data structure that we studied in class. Your algorithm will be graded on correctness, running time efficiency, and clarity.

2) What is the running time of your algorithm regarding the string length N? Use big-Oh notation. Give a brief, concise explanation of why. (Worth 5 points)