

Problem 0. Information (1 point)

Write your name and KAIST ID at the top of **EACH** page on your answer paper. (worth 1 point)

Problem 1. (14 points—7 points @ each)

For **EACH** of the functions below,

1. Give the order of growth in terms of n , using the big-Oh notation. (worth 2 points)
2. Explain why it takes that running time. (worth 5 points)

1)

```
public static void prob1(int n) {  
    int sum=0;  
  
    if(n==0) return;  
    for(int i=0; i<n; i++)  
        sum++;  
    prob1(n/2);  
    prob1(n/2);  
}
```

2)

```
public static void prob2(int n) {  
    int num=0;  
  
    for(int i=n; i>1; i/=2)  
        num++;  
}
```

Problem 2. (20 points)

For **EACH** of the problems below, answer the following questions:

- Using the 64-bit memory cost model from lecture, how much memory does the following code use?
- Give ALL intermediate steps that show how you compute the memory usage.

1) (6 points)

```
Integer[] temp = new Integer[n];
```

2) (14 points)

```
Double[] data = new Double[n];  
for(int i=0; i<n; i++)  
    data[i] = new Double(Math.random());
```

*Note: The memory used by an object of Double type is computed from the following code:

```
public class Double{  
    private double x;  
}
```

Problem 3. (30 points)

For **EACH** of the following questions:

- Select the **worst-case** running time of each problem among the order-of-growth terms in the box below.
- Also, justify your selection.

constant, logarithmic, linear, linearithmic, quadratic, cubic, exponential

- 1) Number of equality tests to insert N keys into an empty linear probing hash table of size $2N$. Assume that we do not resize the hash table. (worth 10 points)
- 2) Running time when we insert an element to be the last element in a dynamic array that has N elements using a doubling-and-halving strategy. (worth 10 points)
- 3) Running time when we perform a push operation on a stack implemented with a singly linked list, which keeps references to both head and tail. (worth 10 points)

Problem 4. (15 points)

You are given an array X that has the following integers. Array indices start at 1, as we did in class.

i	0	1	2	3	4	5	6
X[i]	-	6	5	3	7	2	4

- 1) Is the given array X a binary heap? (worth 3 points)
- 2) Is it possible to make X either a max-heap or min-heap by swapping two integers once? (worth 1 point)
 - If it is possible to do so, specify such two integers in the array X and draw a resulting binary heap—**Do not give an array-representation of a binary heap**.
 - If it is not possible, justify your answer. (worth 11 points)

Problem 5. (20 points)

You have N books on a single bookshelf. Books are stored in descending order by height. The height of each book is distinct, and the height difference is visually comparable. You are required to efficiently sort books in ascending order by height from the shortest to the tallest. In doing so, you should consider the given scenarios below.

For **EACH** of the following problems,

- A. Specify which sorting algorithms you would use.
 - **Note:** You should specify one of the sorting algorithms that we studied in class.
- B. Justify your choice regarding why that sorting algorithm is **best suited** to the given scenario **regarding time and space complexity**.

- 1) The only way to compare the height of two books is by taking them off the bookshelf and compare them. You are required to only take two books from the shelf at one time, then putting them back in the same or swapped order. (10 points)

- 2) At each iteration, you are required to find the remaining shortest book and exchange two books at one time. Another requirement is that you should make a linear number of book exchanges to sort N books in ascending order by height. (10 points)