CS 206: FINAL EXAM

Worth 400 points

- **The total score in this exam paper is 100 points.**
- **100 points will be converted to 400 points by multiplying your earned score by 4.**
- **Please write your answers legibly[1] and neatly. If it is not possible to grade, you will get 0 points.**

#0. INFORMATION (WORTH 2 POINT)

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

#1. (WORTH 11 POINTS)

Consider the following winter() method:

```
public static int winter(int a, int b){
        if (a%b == 0)
                return b;
        else
                return winter(b, a%b);
}
```

1) What does winter(17, 3) return? (0.5 points)
2) Show all intermediate steps that produce the return value you specified in #1-1). (6 points)

3) What does winter(3, 9) return? (0.5 points)
4) Show all intermediate steps that produce the return value you specified in #1-3). (4 points)

#2. (WORTH 15 POINTS)

Suppose that we use a hash table with a dynamic array. Collisions are handled by linear probing. There are N keys in the hash table. Now, we double the size of array and **re-hash those N keys into a new hash table.**

1) What is the worst-case running time of **the re-hash operation specified above?** Use the Big-Oh notation. Give your answer in terms of N. (worth 5 points)
2) Explain why it takes that running time in your answer to #2-1). (worth 10 points)

Continue to next page

[1] Legible writing is clear enough to read.

#3. (WORTH 20 POINTS; 10 POINTS @ EACH)

Consider the following code.

```
public class BinaryTree {
        private int value;
        private BinaryTree left, right;

        public BinaryTree(int v){
                value=v;
                left=null; right=null;
        }

        public void insert(int v){
                if (v<=value){
                        if (left==null) left=new BinaryTree(v);
                        else left.insert(v);
                }
                else {
                        if (right==null) right=new BinaryTree(v);
                        else right.insert(v);
                }
        }

        public static void main(String[] args){
        BinaryTree root=null;

        int[] input = {50, 3, 5, 69, 80, 22, 44, 3};

        for(int i=0;i<input.length;i++) {
                if (root==null)
                        root=new BinaryTree(input[i]);
                else       root.insert(input[i]);
                }

        }
}
```

After we insert 8 integers 50, 3, 5, 69, 80, 22, 44, 3 in that order in the code above, we will have a tree, and we call it the resulting tree T.

1) Give the preorder traversal of the resulting tree T after we insert 8 integers in that order in the code above.
2) Give the inorder traversal of the resulting tree T after we insert 8 integers in that order in the code above.

#4. (WORTH 20.5 POINTS)

A store keeps the information about N customers. You are given two tasks:

I. You are required to store each customer's ID number and the amount of money each customer has spent in the store.
II. You must also provide an efficient way to print all customers' ID numbers in ascending order.

⌘ Note: You are not allowed to use sorting algorithms.

For the given two tasks above (I & II) (see page 2),

1) Specify the ADT that best suits both tasks given above. Specify one of the ADTs we studied in class. (worth 2.5 points)
2) Specify the most efficient[2] data structure that implements the ADT for both tasks given above. (Notes: Specify one of the data structures we have studied in class.) (worth 6 points)
3) Consider inserting one customer's ID number and the amount of money that customer has spent in the store: What is the worst-case running time of this insert operation based on the most efficient data structure? Use the Big-Oh notation in terms of N. (worth 6 points)
4) Consider printing all customers' ID numbers in ascending order: What is the worst-case running time of this print operation based on the most efficient data structure? Use the Big-Oh notation in terms of N. (worth 6 points)

#5. (WORTH 31.5 POINTS; 10.5 POINTS @ EACH)

You are given an array that contains the following 7 equal keys. (Note: the subscript[3], such as 0, 1, 2, 3, 4, 5, 6, is not part of the key. We use the subscript to uniquely identify each of the equal keys).

$A_0 \quad A_1 \quad A_2 \quad A_3 \quad A_4 \quad A_5 \quad A_6$

Give the result of final sorted output for the given 7 equal keys above, using each of the following sorting algorithms we studied in class.

Notes:

- Please do not make any typos.
- **Please write the subscript of each key clearly.**
- If it is not possible to grade, you will get 0 points.

Sorting algorithms	Final sorted output
1) Insertion sort	
2) Selection sort	
3) Top-down mergesort	

[2] We analyze the efficiency regarding its running time.
[3] A subscript is a letter written below the line. For instance, in "H_2", 2 is a subscript.