HOMEWORK BEFORE MIDTERM EXAM

(1) Prove that for any positive integer k, $\sum_{i=1}^{n} i^k \lg i = O\left(n^{k+1} \log n\right)$

(2) Describe the worst-case running-time function of the following algorithms:

```
procedure Problem2a(n)
    s := 0
    for i := 0 to n do
        for j := 1 to i do
            for k = j to i + j do
                s := s + 1
            end for
        end for
    end for
end procedure
```

```
procedure Problem2b(n)
    i := 0
    j := n
    s := 0
    while i < j do
        i := i + 2
        j := j − 3
        s := s + 1
    end while
end procedure
```

```
procedure Problem2c(n)
    if n ≤ 0 then return
    end if
    for i = 1 to n do
        for j = 1 to n do
            s := s + 1
        end for
    end for
    for i = 1 to 4 do
        Problem2c(n/4)
    end for
end procedure
```

(3) Given n arrays, each array contain n positive integers. Write an $O\left(n^2 \log n\right)$ algorithm to find the smallest n sums out of n^n possible sums that can be obtained by picking one positive integer from each of n arrays. For example, given three arrays as follows: $[5, 1, 8]$, $[5, 2, 9]$, and $[6, 7, 10]$. The smallest n sums of the given array is $[9, 10, 12]$.

(4) Write an algorithm to sort n integers in the range 0 to $n^3 - 1$ in $O(n)$ time.

(5) Assuming that there are no duplicate keys in a binary search tree, develop an algorithm to determine the pre-order traversal of the tree based on their

in-order and post-order traversal. For example,
In-order traversal: 3 1 4 0 2 5
Post-order traversal: 3 4 1 5 2 0
Output: 0 1 3 4 2 5

(6) Write the algorithm TREE-DELETE(T, z) to delete a node z from a Binary Search Tree. When z has two children, use its PREDECESSOR (maximum in the left subtree) to replace it instead of the SUCCESSOR.

(7) The d-ary heap, also known as the d-heap, is a type of priority queue data structure. It extends the concept of the binary heap by allowing nodes to have d children instead of just 2.
   (a) Describe how to represent a d-ary heap in an array.
   (b) Give an efficient implementation of EXTRACTMIN in a d-ary min-heap. Analyze its running time in terms of d and n.
   (c) Give an efficient implementation of DECREASEKEY in a d-ary min-heap. Analyze its running time in terms of d and n.
   (d) Give an efficient implementation of INSERT in a d-ary min-heap. Analyze its running time in terms of d and n.