INDIAN STATISTICAL INSTITUTE

ENDTERM EXAMINATION
M.TECH(CrS) I YEAR

PROGRAMING AND DATA STRUCTURES

Date: 17.11.2025 Maximum marks: 100 Duration: 3.5 hours.

The paper contains 115 marks. Answer as much as you can, the maximum you can score is 100.

1. Give the tightest possible upper bound for the worst case running time for each of the following operations in terms of n. Give a very short explanation for your answer.

   (a) Pushing an element in a stack containing n elements. The stack is implemented using re-sizeable arrays.

   (b) Printing out the values in an AVL tree in post order.

   (c) Finding the maximum value in a binary min heap of size n.

   (d) Moving the values from a binary min heap, into an initially empty array of the same size. The final contents of the array should be sorted from low to high.

   (e) Given a binary search tree containing n integers, create an AVL tree containing the same values. You should not destroy the original binary search tree in the process. $[5 \times 2 = 10]$

2. Consider the Quack abstract data type which is a combination of a queue and a stack. A Quack essentially maintains a list with the following operations:

   `QPush(`x`):` Add x at the front of the list.

   `QPop():` Remove and return the element in the front of the list.

   `QPull():` Remove the element in the back of the list.

   Describe an implementation of a Quack using three stacks and $O(1)$ additional memory, so that the amortized time for any `QPush`, `QPop`, or `QPull` operation is $O(1)$. In particular, each element in the Quack must be stored in exactly one of the three stacks. You cannot access the component stacks except through the interface functions of the stacks, i.e., Push and Pop which runs in $O(1)$ time. Your solution should include the following.

   (a) A high level description of your implementation in English.

   (b) Pseudocodes for the QPush(x), QPop and QPull operations.

   (c) A precise argument which proves that the amortized cost of each operation is $O(1)$.

   $[5 + 10 + 10 = 25]$

3. A d-ary heap is like a binary heap but with one possible exception, non leaf nodes have d children instead of two children.

   (a) Discuss an array implementation of a d-ary heap. In particular, write procedures for finding the children and parent of a given node.

   (b) What is the height of a d-ary heap of n elements in terms of n and d.

   (c) Give an efficient procedure to implement Extract-Max in a d-ary max-heap. Analyze the running time in terms of d and n.

   [2 + 2 + 6 = 10]

4. In an initially empty binary search tree, the following elements are inserted in the given order: 3, 1, 10, 12, 4, 20, 14, 19, 22, 18, 2, 8

   (a) Draw the resulting tree.

   (b) Is the tree balanced? If the tree is not balanced, which nodes are not balanced, mark them in your answer to part (a).

   (c) In the tree of part (a) delete the nodes containing 12, 3, 10 in the given order and draw the resulting tree. [5 + 5 + 5 = 15]

5. Describe an implementation of a Min-Max Priority Queue ADT with the following operations:

   - Insert(x): Inserts x in the data structure.
   - ExtractMin(): Deletes and return the minimum
   - ExtractMax(): Deletes and return the maximum
   - FindMin(): Find the minimum
   - FindMax(): Find the maximum

   Your implementation should take $O(1)$ time for FindMin()and FindMax() and $O(\log(n))$ time for all other operations where n is the number of elements in the data structure. [10]

6. (a) Define a universal hash family.

   (b) Let p be a prime. Define $\mathcal{X} = \mathbb{Z}_p^2 \cup \mathbb{Z}_p^3$. We define a family of hash functions $\mathcal{H} = \{H_K\}_{K \in \mathbb{Z}_p}$ where for each $K \in \mathbb{Z}_p$, $H_K : \mathcal{X} \to \mathbb{Z}_p$ is defined as follows. If $x = (x_1, x_2) \in \mathbb{Z}_p^2$, then

   $$H_K(x) = x_1 K + x_2 K^2 + 2K^3 \bmod p,$$

   and if $x = (x_1, x_2, x_3) \in \mathbb{Z}_p^3$,

   $$H_K(x) = x_1 K + x_2 K^2 + x_3 K^3 + 3K^4 \bmod p.$$

   Find the collision probability of $\mathcal{H}$. Is $\mathcal{H}$ a universal family?

(c) Let $\mathcal{H}$ be an universal family of hash function where each $h \in \mathcal{H}$ maps U to $\{0, 1, \ldots, m-1\}$, where U is the set of all possible keys and m a fixed positive integer. Let T be a hash table (with chaining) which is constructed using a hash function uniformly selected from $\mathcal{H}$, thus T has m slots. Suppose T contains n elements and let $\alpha = n/m$. Prove that for any key $k \in U$, the expected length of the chain in T to which k is hashed is at most $(1 + \alpha)$.

[2 + (10 + 2) + 8 = 20]

7. You are required to store a set of IP addresses (recall that an IP address can be seen as a vector of four bytes written as $x = x_1.x_2.x_3.x_4$, where each x_i can be treated as an integer between 0 and 255). The operations required are Find(x) and Insert(x).

   (a) Describe an implementation of a randomized data structure where both Find(x) and Insert(x) runs in $O(1)$ expected time. Your description should include all details including the run time analysis.

   (b) Describe another implementation where both Find(x) and Insert(x) runs in $O(1)$ worst case time. This implementation should take $O(k)$ space when k many IP addresses are stored.

   (c) Modify the implementation in (b) to incorporate another operation MostFrequent(), which returns the most frequently accessed IP address so far. The operation MostFrequent() should run in constant time.

[10 + 10 + 5 = 25]