BRAC UNIVERSITY
Department of Computer Science and Engineering

CSE220: Data Structures

A

Examination: Final
Duration: 1 Hour 40 Minutes

Semester: Fall 2024
Full Marks: 35
No. of Pages: 5

Name: (Please write in CAPITAL LETTERS)	ID:	Section:

✔ **Answer all questions. No washroom breaks. Understanding questions is part of the Exam.**
✔ At the end of the exam, put the question **paper** inside the answer script and **return both**.

Question 1: CO1 [7 Points]

Write the correct answer (only option) in your answer script:

i. The preorder traversal sequence of a binary search tree is 30, 20, 10, 15, 25, 23, 39, 35, 42. Which of the following is the postorder traversal sequence of the same tree?

A. 10, 20, 15, 23, 25, 35, 42, 39, 30
B. 15, 10, 25, 23, 20, 42, 35, 39, 30
C. 15, 20, 10, 23, 25, 42, 35, 39, 30
D. 15, 10, 23, 25, 20, 35, 42, 39, 30

ii. Suppose that we have numbers between 1 and 100 in a binary search tree and want to search for the number 55. Which of the following sequences CANNOT be the sequence of the nodes examined?

A. {10, 75, 64, 43, 60, 57, 55}
B. {90, 12, 68, 34, 62, 45, 55}
C. {9, 85, 47, 68, 43, 57, 55}
D. {79, 14, 72, 56, 16, 53, 55}

iii. The following keys are inserted into a hashtable in which collision resolution is done by chaining. If the hash function is h(k) = k mod 4, what is the length of the longest collision chain if the following keys are inserted sequentially 1, 5, 28, 17, 15, 19? The length of the hashtable is 4.

A. 1
B. 2
C. 3
D. 4

iv. Which of the following hash functions is suitable for a hash table, where **k** represents the key and **m** represents the size of the hash table?

A. h(k) = k×m
B. h(k) = (k+m) mod 29
C. h(k) = (k+9) mod m
D. h(k) = (k mod m)+3

v. If we have two different Binary Trees with the same in-order traversal. Which of the following is correct?

A. They must be identical trees
B. They must have the same height
C. They must have the same preorder traversal
D. They can have different structures

vi. Which one of the following hash functions on integers will distribute keys most uniformly over 10 buckets numbered 0 to 9 for i ranging from 0 to 2020?

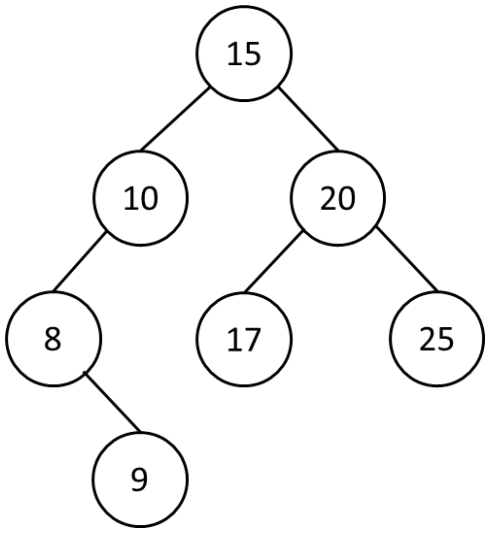
A. h(i) =i^2 mod 10
B. h(i) =i^3 mod 10
C. h(i) = (11 ∗ i^2) mod 10
D. h(i) = (12 ∗ i) mod 1

vii. In a hash table with forward chaining, which data structure is used to store multiple elements at the same index?

A. Array
B. Linked List
C. Binary Tree
D. Stack

Question 2: CO3 [2 + 1 + 1 Points]

Consider the following BST.

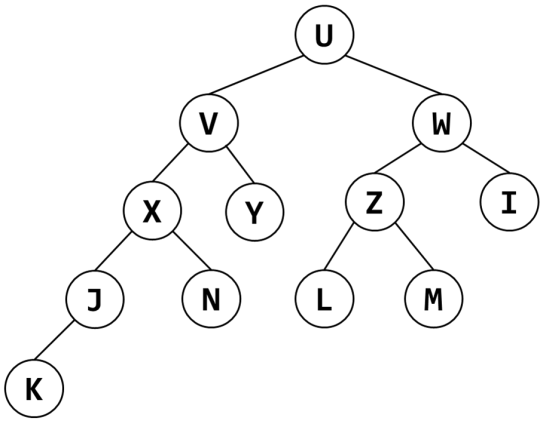


A. Delete the root 15 with its predecessor and draw the resulting tree.
B. Is the tree drawn on A a complete binary tree (Yes / No)?
C. Is the tree drawn on A a balanced binary tree (Yes / No)?

Question 3: CO4 [8 Points]

You are given a binary tree where each node stores a character. You need to generate a key based on the leaf nodes positioned at the even levels **from right to left**. Your task is to implement a **function/method** named **build_key** that takes the root of the binary tree and constructs the key string based on the criteria above and finally returns it. If there are no leaf nodes on even levels, then the function/method should return an empty string.

Consider the Node class for Binary Tree already defined with elem (Type: String), left (Type: Node), and right (Type: Node) variables. **You can use helper functions or extra parameters in the given function if required. You can not use any other data structures than the given binary tree.**

Input tree	Returned String	Explanation
	**IYK**	All the leaf nodes from right to left are: **I, M, L, Y, N, K** However, only **I, Y, K** are on even levels. Hence the output string is **"IYK"**

Python Notation	Java Notation

`def build_key(root):`
`  #To do`
	`static String build_key(Node root){`
	`  //To do`
	`}`

Question – 4: CO3 [8 Points]

As a Software Engineer at Microsoft, you need to schedule tasks on the CPU based on priority. Given n tasks with unique priorities in an array and a value k, perform the first k tasks with the highest priority.

Implement the function **`cpu_scheduler(tasks, k)`** to print the task numbers of the top **`k`** tasks in descending order of priority. Use the provided MaxHeap or MinHeap classes with **`insert(item)`** and **`extract()`** (extract will return the extracted value) methods. **You cannot iterate through the array manually to find out the highest priority task. Check sample input-output to find out the printing format.**

You can create an Empty Heap using the following code	
Python Notation	Java Notation
`max_heap = MaxHeap()` `min_heap = MinHeap()`	`MaxHeap maxHeap = new MaxHeap()` `MinHeap minHeap = new MinHeap()`

Python Notation	Java Notation

`def cpu_scheduler(tasks, k):`
`  #To do`
	`static void cpu_scheduler(int[ ] tasks, int k){`
	`  //To do`
	`}`

Input	Output	Explanation

`tasks = [45, 70, 85, 60, 90, 75]`
`k = 3`
	`Task 1 - Priority 90`
	`Task 2 - Priority 85`
	`Task 3 - Priority 75`
		Here are the 6 tasks given in the array. The value of k is 3 so the function will print the first 3 tasks with higher priority according to the shown format.

Question – 5: CO2 [8 Points]

You are organizing a Networking Conference where participants can interact multiple times, forming an undirected network graph where:

- **Vertices** represent participants.
- **Edges** represent interactions (multiple edges indicate repeated interactions). 2 vertex having multiple edges between them means 2 participants interact multiple times.

A network graph is **complete** if every participant interacts with every other participant at least once.

Write a function **`is_complete`** to determine if a given network graph, represented as an unweighted adjacency list, is complete or not. If complete the function will return True, else False. **The Edge class has source (Type: String), destination (Type: String), and a next (Type: Edge) variable. You can use extra arrays if you want but cannot modify the given adjacency list.**

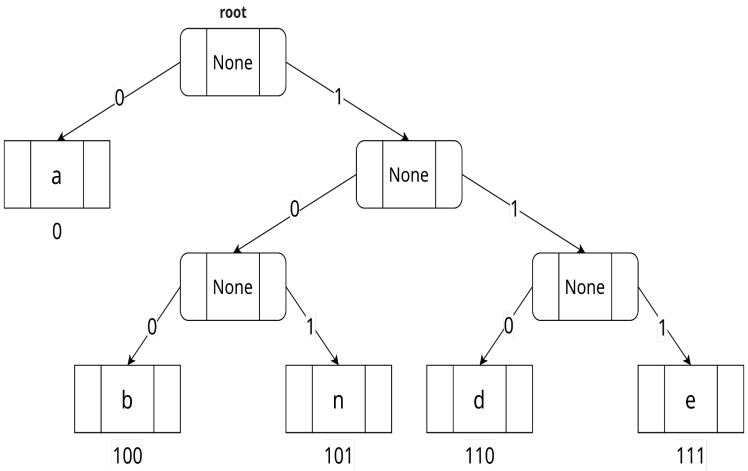
Python Notation	Java Notation

`def is_complete(network):`
`  #To do`
	`static boolean is_complete(Edge[ ] network){`
	`  //To do`
	`}`

Input	Output	Explanation
network = [0: (0, 1) → (0, 2) → (0, 3), 1: (1, 0) → (1, 2) → (1, 3), 2: (2, 0) → (2, 1) → (2, 3), 3: (3, 0) → (3, 1) → (3, 2)]]	`True`	There are 4 participants: 0, 1, 2, and 3. Every participant has met every other participant at least once: ● Participant 0 met with participants 1, 2, and 3. ● Participant 1 met with participants 0, 2, and 3. ● Participant 2 met with participants 0, 1, and 3. ● Participant 3 met with participants 0, 1, and 2. Since every pair of participants has met at least once, the network is **complete**.
network = [0: (0, 1) → (0, 2) → (0, 3), 1: (1, 0) → (1, 2) → (1, 0), 2: (2, 0) → (2, 1) → (2, 3), 3: (3, 1) → (3, 2) → (3, 2)]	`False`	There are 4 participants: 0, 1, 2, and 3. ● Participant 1 has not met with participant 3. ● Participant 3 has not met with participant 0. Since there are missing interactions, the network is **incomplete.**

Question – 6: Bonus [5 Points]

The binary tree, called the Huffman coding tree, has all the possible valid characters as the leaves and all the internal nodes contain None/null. **Each transition to a left child represents a string 0-value bit and each transition to a right child from the parent represents a string 1-value bit. The bit representation of a character is the sequence of 0s and 1s you get through the path from the root to the leaf holding the characters**.



Here, the representation of **a** would be 0. Since we're moving only to the left 1 time. Similarly, **n** would be 101 since we're moving right, left, and right, and **d** would be 110 since we're moving right, right, left.

Each node of the Huffman encoding tree has four properties, a parent property to hold a reference of the parent of the current node, a left and a right to refer to the two children, and finally, a value property which is a character for the leaves and None/Null for the internal nodes.

Your task is to implement the following recursive function that takes the root of a Huffman encoding tree and a string text as parameters and returns a string representing the compressed bit encoding of the input string.

[The parent property of the node is absolutely not necessary to solve the task. But you can optimize the solution of this by using the node's parent property]

Note: You can create other recursive helper functions and call them inside the given input function. You can also use loops in your recursive functions. **There is no restriction on using other data structures in this problem.**

Sample Function Call	Sample Returned String	Explanation
`getEncoding(root,"and")`	"0101110"	Here, the bit representation of strings of "a", "n", "d" are "0", "101", "110" respectively. So, they're concatenated and shown in the output.

Python Notation	Java Notation

`def getEncoding(root, st):`
`  #To do`
	`static String getEncoding(Node root, String st){`
	`  //To do`
	`}`