Ramakrishna Mission Vivekananda Educational and Research Institute

Belur Math, Howrah, West Bengal
Department of Computer Science
Final Term (Semester 1)
MSc DSAI

Course: CS110 (Data Structures and Algorithms) Date: 04 Dec 2025
Instructor: Sri Niladri Banerjee Time: 3 hrs
Student Roll: Max Marks: 100
Student Name:

Instructions:

- The answers should be clear, concise, and complete to earn the full allotted marks. All steps in the problem-solving process should be clearly stated.
- Code should be Python/C/C++. The accuracy of the code logic is important to obtain marks.
- The use of calculators or other electronic devices is prohibited.

Section A **[5 questions × 2 marks = 10 marks]**

---

1. Determine the Θ bounds for the following recurrence relation: $T(n) = 2T\left(\frac{n}{2}\right) + n \log n$
2. Evaluate the prefix expression: `+*23-51`
3. Write the `push` operation code for a stack. Assume the current top element is `10`, a new node with value `20` is to be pushed, and the current stack height is `1`. Begin your implementation with `def push:`.
4. If number of nodes of a binary tree given = 15, Determine the min and max height of a binary tree?
5. Draw a binary tree T that simultaneously satisfies the following conditions:
   - Each internal node of T stores a single character.
   - A preorder traversal of T yields the string `EXAMFUN`.
   - An inorder traversal of T yields the string `MAFXUEN`.

Section B **[(answer any 9 questions) × 10 marks = 90 marks]**

---

1. Cycle Detection in Singly Linked List using Floyd's Algorithm
   (a) Construct a singly linked list with at least 10 nodes. Introduce a cycle by connecting the last node to one of the earlier nodes (not the head). Then, describe (mathematical proof) the Floyd's Cycle Detection algorithm and explain why it is able to detect the presence of a cycle in such a structure. **[5 marks]**

   (b) Implement Floyd's Cycle Detection algorithm in a programming language of your choice. Ensure your code correctly detects the presence of a cycle and identifies the starting node of the cycle. Provide brief comments or explanation to support your implementation. **[5 marks]**

2. (a) A hash table of size 11 uses open addressing. The primary hash function is $h_1(k) = k \bmod 11$ and the secondary hash function is $h_2(k) = 1 + (k \bmod 7)$. The probe sequence is defined as $h(k, i) = (h_1(k) + i \cdot h_2(k)) \bmod 11$ for the i^{th} probe. Keys $63, 50, 25, 79, 67, 24$ are inserted in this order. In which slot does the key 24 get stored? **[5 marks]**
   (b) Explain Primary clustering and secondary clustering with an example for each. **[5 marks]**

3. (a) Construct a min heap by applying the *Heapify* procedure on the following keys (given in level order):

145, 40, 25, 65, 12, 48, 18, 1, 100, 27, 73, 45, 9, 30

using bottom-up heapify. Show the array after each heapify call for indices n/2 down to 1 (only show the array after each index's final sift-down). Final array must be given in level order. (Partial credit for correct intermediate steps.) **[5 marks]**

(b) Derive the time complexity of the Heapify operation. **[5 marks]**

4. (a) Consider the Quick Sort algorithm applied on an array of n distinct elements. If the pivot chosen at each step always ends up being the largest element of the current subarray, derive the recurrence relation for the worst-case time complexity and solve it. **[5 marks]**

(b) Suppose the pivot is always chosen as the median of the current subarray. Derive the recurrence relation for the best-case time complexity and solve it. [Hint: In order to partition the entire array, you need to scan n elements, which is its partitioning cost] **[5 marks]**

5. (a) Explain the Selection sort process with the array = 8,6,3,2,5,4. Results of 1 pass is sufficient to explain the concept. Describe the time complexity for Total comparisons and swaps. **[5 marks]**

(b) Write the algorithm for Selection Sort **[5 marks]**

6. (a) Draw an AVL Tree with keys = 10,20,30,25,28,27,5. Draw the diagram for each element insert. Write the balance factors for each node and write the rotation that is required during each step if applicable.[5 marks]

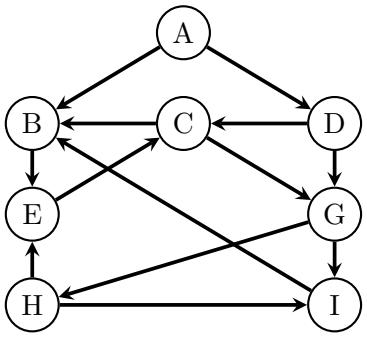
(b) Draw a non-height balanced binary tree with the above keys and highlight the difference in their time complexities. **[5 marks]**

7. (a) Generate BFS traversal for Graph 1. Mention nodes visited, neighbours, queue in table format. Mention the BFS traversal path. **[5 marks]**

(b) Identify Tree edges, Cross edges, Back edges and Forward edges whatever is applicable in BFS Spanning Tree. **[5 marks]**

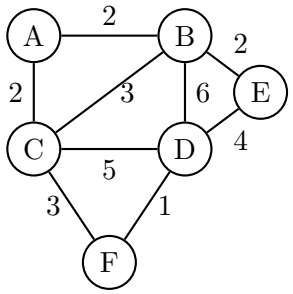
8. (a) Generate DFS traversal for Graph 1. Mention nodes visited, neighbours, stack in table format. Mention the DFS traversal path **[5 marks]**

(b) Identify Tree edges, Cross edges, Back edges and Forward edges whatever is applicable in DFS Spanning Tree **[5 marks]**



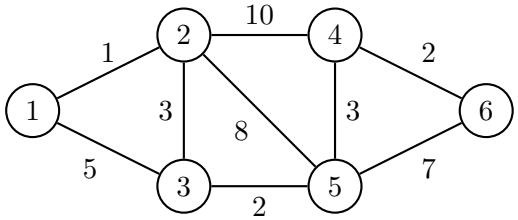
Graph 1

9. Find the min cost of spanning tree generated using each of Kruskal and Prim algorithms from Graph 2. Mention the edges selected in each of these processes in a table. **[10 marks]**



Graph 2

10. Starting with source vertex = 1, find the minimum distances to all vertices using Dijkstra's algorithm on Graph 3. Highlight the relaxation of nodes logic and show the step-by-step process in a table **[10 marks]**



Graph 3

11. Write the `delete_node` code for Binary Search Tree **[10 marks]**

```
def __delete_node(self, current_node, value):
    if current_node is None:
        return None
    # Write your code
    return current_node

def delete_node(self, value):
    return self.__delete_node(self.root, value)
```


End of exam