B.Sc. (Hons.) Computer Science III Semester (NEP)

Data Structures Guidelines

S. No.	Topic	Reference	Contents
1	Unit 1 - Growth of Functions, Recurrence Relations	[1] [2]	Ch-4 4.1, 4.2: 4.2.1-4.2.5 Ch-4: 4.3, 4.4, 4.5
2	Unit 2 - Arrays, Linked Lists, Stacks, Queues, Deques	[1] [1]	Ch-3: 3.1 (till page 114 – excluding tic-tac-toe) 3.2, 3.3, 3.4 ch-5: 5.1, 5.2, 5.3: 5.3.1-5.3.3
3	Unit 3 - Recursion	[1]	ch-3: 3.5 upto page 135, 3.5.1, 3.5.2 ch-4: 4.2.6
4	Unit 4 - Trees, Binary trees	[1]	ch-7: 7.1, 7.2, 7.3.1-7.3.4, 7.3.6 upto page 299
5	Unit 5 - Binary Search Trees, Balanced Search Trees	[1]	ch-10: 10.1, 10.2 upto 10.2.1 (10.2.2 to be covered for practicals only)
6	Unit 6 - Binary Heap	[2]	ch-6: 6.1-6.3

References

1. Goodrich, M.T, Tamassia, R., & Mount, D., Data Structures and Algorithms Analysis in C++, 2nd edition. Wiley, 2011.
2. Cormen, T.H., Leiserson, C.E., Rivest, R. L., Stein C. Introduction to Algorithms, 4th edition, Prentice Hall of India, 2022.

Additional References

(i) Sahni, S., Data Structures, Algorithms and applications in C++, 2nd edition, Universities Press, 2011.
(ii) Langsam Y., Augenstein, M. J., & Tanenbaum, A. M. Data Structures Using C and C++, Pearson, 2009.

Practicals List

1. Write a program to implement singly linked list as an ADT that supports the following operations:
   i. Insert an element x at the beginning of the singly linked list
   ii. Insert an element x at i^{th} position in the singly linked list
   iii. Remove an element from the beginning of the doubly linked list
   iv. Remove an element from i^{th} position in the singly linked list.
   vi. Search for an element x in the singly linked list and return its pointer

2. Write a program to implement doubly linked list as an ADT that supports the following operations:
   i. Insert an element x at the beginning of the doubly linked list
   ii. Insert an element x at the end of the doubly linked list
   iii. Remove an element from the beginning of the doubly linked list
   iv. Remove an element from the end of the doubly linked list

3. Write a program to implement circular linked list as an ADT which supports the following operations:
   i. Insert an element x in the list
   ii. Remove an element from the list
   iii. Search for an element x in the list and return its pointer

4. Implement Stack as an ADT and use it to evaluate a prefix/postfix expression.
5. Implement Queue as an ADT.
6. Write a program to implement Binary Search Tree as an ADT which supports the following operations:
   i. Insert an element x
   ii. Delete an element x
   iii. Search for an element x in the BST
   iv. Display the elements of the BST in preorder, inorder, and postorder traversal

7. Write a program to implement insert and search operation in AVL trees.