Guidelines of Generic Elective Semester IV

(NEP UGCF 2022)

GE 4a: Data Structures Using Python

(Effective from Academic Year 2024-25)

S. No.	Topic	Contents	Reference	Hours
1	Unit 1 - Growth of Functions, Recurrence Relations	Ch-3 3.2, 3.3 Ch-4: 4.3, 4.4, 4.5	[1] [2]	8
2	Unit 2 - Arrays, Linked Lists, Stacks, Queues, Deques	5.6 (excluding tic-tac toe) Ch-6 (6.1, 6.2, 6.3) Ch-7: 7.1, 7.2, 7.3	[1] [1]	16
3	Unit 3 - Recursion	Ch-4: 4.1 (4.1.1 and 4.1.3), 4.4 (4.4.1 and 4.4.2)	[1]	4
4	Unit 4 - Trees, Binary trees, Binary Search Trees, Balanced Search Trees	Ch-8: 8.1, 8.2, 8.3 (8.3.1), 8.4 (upto 8.4.4) Ch-11: 11.1, 11.2 (introduction), 11.3 upto 11.3.1 (11.3.2 to be covered for practicals only)	[1]	13
5	Unit 5 - Binary Heap	ch-6: 6.1-6.3	[2]	4

References

1. Goodrich M.T., Tamassia R., & Goldwasser M.H., Data Structures and Algorithms in Python, Wiley, 2021.
2. Cormen, T.H., Leiserson, C.E., Rivest, R. L., Stein C. Introduction to Algorithms, Prentice Hall of India, 2022.
3. Taneja, S. and Kumar, N., Python Programming: A modular approach, Pearson, 2017.

Additional References

(i) Drozdek A., Data Structures and Algorithms in Python, 1 st Edition, Cengage learning, 2024.

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.