GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Engineering
Level: Diploma
Branch: Information Technology
Course / Subject Code : DI03016031
Course / Subject Name :Data Structures with Python

w. e. f. Academic Year:	2024-25
Semester:	3rd
Category of the Course:	PCC

Prerequisite:	Basic knowledge of Python (variables, loops, functions). Familiarity with fundamental programming concepts.
Rationale:	Development of application systems and software that use underlying architecture of machines efficiently and effectively requires the ability to use and manipulate various types of Data Structures and other constructs. This being a fundamental ability which is language neutral yet requires use of a language for its implementation. As far as data structures are concerned, the course covers Python dictionaries as well as classes and objects for defining user defined data types such as linked lists and binary search trees. This course is designed to develop an integrated ability to efficient software development and apply the knowledge to various application systems; hence this course is very important for IT diploma engineers.

Course Outcome:
After Completion of the Course, Student will able to:

No	Course Outcomes	RBT Level
01	Understand linear and non-linear data structures.	Apply
02	Implement Object Oriented Programming concepts in Python.	Apply
03	Implement basic data structures such as stacks, queues.	Apply
04	Develop and manipulate linked lists with various operations.	Apply
05	Apply Algorithms for solving problems like searching and sorting of data.	Apply
06	Implement nonlinear data structures like trees.	Apply

**Revised Bloom's Taxonomy (RBT)*

Teaching and Examination Scheme:

Teaching Scheme (in Hours)			Total Credits L+T+ (PR/2)	Assessment Pattern and Marks				Total Marks
				Theory		Tutorial / Practical		
L	T	PR	C	ESE (E)	PA(M)	PA(I)	ESE (V)	
3	0	2	4	70	30	20	30	150

Course Content:

Unit No.	Content	No. of Hours	% of Weightage
1.	**Basic Concepts of Data Structures** 1.1 Data Structure Basic Concepts 1.2 Types of data structures 1.3 Analysis Terms (for the definitions purpose only) : Time Complexity, Space Complexity, Asymptotic Notations , Big 'O', Notation, Best case Time, Complexity, Average case Time Complexity, Worst case, Time Complexity 1.4 Python Specific Data, Structures-List, Tuple, Set, Dictionary 1.5 Array in Python, import array, import numpy, Operations on Arrays, Arrays vs List	**04**	**9**
2.	**Basics of Object-Oriented Programming** 2.1 Oops Concepts 2.2 Class and Object 2.3 Constructors 2.4 Types of methods Instance method, Class method, static method 2.5 Data Encapsulation 2.6 Inheritance - single, multiple, multi-level, hierarchical, hybrid 2.7 Polymorphism through inheritance 2.8 Abstraction - abstract class	**08**	**17**

3.	**Stack and Queues** 3.1 Overview of Stack 3.2 Operations on Stack - Push, Pop 3.3 Implementation of Stack using List 3.4 Application of Stack - Infix, Prefix and Postfix Forms of Expressions, Evaluations of postfix expression, Recursive Functions (factorial, Fibonacci series) 3.5 Overview of Queue 3.6 Operations on Queue - Enqueue and Dequeue 3.7 Implementation of Queue using List 3.8 Limitation of Single Queue 3.9 Concepts of Circular Queue 3.10 Application of queue 3.11 Differentiate circular queue and simple queue	**08**	**18**
4.	**Linked List** 4.1 Overview of Linked list 4.2 Types of Linked List 4.3 Basic operations on singly linked list : Insertion of a new node in the beginning of the list, at the end of the list, after a given node, before a given node, Deleting the first and last node from a linked list, Count the number of nodes in linked list. 4.4 Overview of circular linked list 4.5 Difference between circular linked list and singly linked list 4.6 Overview of doubly linked list 4.7 Applications of linked list	**09**	**20**
5.	5.1 Searching an element into List: Linear Search, Binary Search 5.2 Sorting Methods: Bubble Sort, Selection Sort, Quick Sort, Insertion Sort, Merge Sort	**09**	**20**

6	6.1 Binary trees: Complete Binary Tree, Basic Terms: level number, degree, in-degree and out-degree, leaf node 6.2 Binary Search Tree: Insertion of a node in binary tree, Deletion of a node in binary tree, Searching a node in binary tree 6.3 Tree Traversal : Inorder, Preorder, Postorder 6.4 Applications of binary tree	**07**	**16**
	Total	**45**	**100**

Suggested Specification Table with Marks (Theory):

Distribution of Theory Marks (in %)					
R Level	**U Level**	**A Level**	**N Level**	**E Level**	**C Level**
11	**30**	**22**	**-**	**-**	**-**

Where R: Remember; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create (as per Revised Bloom's Taxonomy)

References/Suggested Learning Resources:

(a) Books:

Sr No.	Title of Book	Author	Publication with place, year and ISBN
1	Data structures and algorithms in python	M.Goodrich	Wiley, 2013 ISBN: 978-1-118-29027-9
2	Data Structures and Algorithmic Thinking with Python	N.Karumanchi	Career Monk Publications, 2016 ISBN:978-81-921075-9-2
3	Core Python Programming	Wesley J. Chun	Prentice Hall, ISBN: 978-0-13-226993-3
4	Data Structures and Algorithms Using Python	R. Necaise	John Wiley & Sons, 2011 ISBN: 978-0470618295
5	Python Programming	N.Fatak, S.Chavda	Mahajan Publication,2021 978-93-93218-00-1
6	Advanced Python Programming	S.Chawda, P.Chavda	Mahajan Publication,2022 978-93-93218-22-3

(b) Open source software and website:

1. VisuAlgo-Python – A tool for visualizing data structures and algorithms.
2. Algopy – A collection of Python implementations of common algorithms and data structures.
3. PyDSA – A Python library for data structures and algorithms.
4. https://onlinecourses.nptel.ac.in/noc22_cs26/preview

5. https://ds1-iiith.vlabs.ac.in/Introduction.html
6. https://cse01-iiith.vlabs.ac.in/Introduction.html
7. https://www.cs.usfca.edu/~galles/visualization/Algorithms.html

Suggested Course Practical List:

The following practical outcomes (PrOs) are the subcomponents of the COs. These PrOs need to beattained to achieve the COs.

S. No.	Practical Outcomes (PrOs)	Unit No.	Approx. Hrs. required
1.	Write a program to read a list of elements. Modify this list so that it does not contain any duplicate elements, i.e., all elements occurring multiple times in the list should appear only once.	I	02
2.	Implement a Program for two matrix multiplication using simple nested loop and NumPy module.	I	02
3.	Design an employee class for reading and displaying the employee information, the getInfo() and displayInfo() methods will be used respectively. Where getInfo() will be a private method.	II	02
4.	Design a class Complex for adding the two complex numbers and also show the use of constructor.	II	02
5.	Design a class for single level, multiple and hierarchical inheritance using public and private type derivation.	II	02
6.	Write a Python program to demonstrate method overriding using inheritance.	II	02
7.	Implement a program to convert infix notation to postfix notation using stack.	III	02
8.	Implement a program to implement QUEUE using list that performs following operations: ENQUEUE, DEQUEUE, DISPLAY	III	02
9.	Implement program to perform following operation on singly linked list: a. Insert a node at the beginning of a singly linked list. b. Insert a node at the end of a singly linked list. c. Insert a node after the given node of a singly linked list. d. Insert a node before the given node of singly linked list. e. Delete a node from the beginning of a singly linked list. f. Delete a node from the end of a singly linked list. g. Count the number of nodes of a singly linked list. h. Display content of singly linked list	IV	04

S. No.	Practical Outcomes (PrOs)	Unit No.	Approx. Hrs. required
10.	Implement a python program to search a particular element from list using Linear and Binary Search.	V	02
11.	Implement Bubble sort,Selection sort, Insertion sort and Merge sort algorithm.	V	04
12.	Write a menu driven program to perform following operation on Binary Search Tree: a. Create a BST. b. Insert an element in BST. c. Pre-order traversal of BST. d. In-order traversal of BST. e. Post-order traversal of BST. f. Delete an element from BST	VI	04
			30 Hrs.

List of Laboratory/Learning Resources Required:

Sr No.	Equipment Name with Broad Specifications	PrO. No.
1	Computer system with operating system: Windows 7 or higher Ver., macOS, and Linux, with 4GB or higher RAM, Python versions: 2.7.X, 3.6.X or higher	All
2	Python IDEs and Code Editors Open Source : IDLE, Jupyter	

Suggested Project List:

A suggestive list of projects is given here. This has to match the COs. Similar projects could be added by the concerned course teacher:

Maze Solver using Breadth-First Search (Queue)

Implement a maze-solving application using BFS, where the maze is represented as a graph. The program finds the shortest path from the entry to the exit, making it useful in robotics and game development.

a) File Version Control using Stack

Develop a version control system where each change to a file is stored in a stack (LIFO). Users can revert to previous versions by popping changes from the stack. This mimics how undo/redo functionalities work in text editors and software development tools.

b) Student Records Management using Binary Search Tree (BST)

This project allows users to manage student records efficiently using a BST. Each node represents a student, storing details like name, ID, and marks. The BST enables quick searching, insertion, and deletion of records based on student IDs. Additional features can include sorting students based on marks and displaying the top performers.

Suggested Activities for Students:

Other than the classroom and laboratory learning, following are the suggested student related co curricular activities which can be undertaken to accelerate the attainment of the various outcomes in this course:

a) Develop a DS-based Application – Build projects like a contact book (linked lists), task scheduler (heaps), or inventory system (queues & stacks) using advanced Python concepts.
b) Explore Libraries – Use NumPy, pandas, or NetworkX to analyze and manipulate data structures efficiently.
c) API Integration – Fetch and process real-world data (e.g., weather, stock prices) using REST APIs and store them in hash tables or graphs.
d) Competitive Coding – Solve data structure challenges on LeetCode, HackerRank, or Codewars to enhance problem-solving skills.

* * * * * * *