GUJARAT TECHNOLOGICAL UNIVERSITY (GTU)

Competency-focused Outcome-based Green Curriculum-2021 (COGC-2021)
Semester-III

Course Title: Data Structure with Python
(Course Code: 4331601)

Diploma programme in which this course is offered	Semester in which offered
Information Technology	Third

1. 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.

2. COMPETENCY

The purpose of this course is to help the student to attain the following industry identified competency through various teaching learning experiences:

- **Implement various types of data structures algorithms using python.**

3. COURSE OUTCOMES (COs)

The practical exercises, the underpinning knowledge and the relevant soft skills associated with the identified competency are to be developed in the student for theachievement of the following COs:

a) Understand linear and non-linear data structures.
b) Implement Object Oriented Programming concepts in Python.
c) Implement basic data structures such as stacks, queues and linked lists.
d) Apply Algorithms for solving problems like searching and sorting of data.
e) Implement nonlinear data structures like trees.

4. TEACHING AND EXAMINATION SCHEME

Teaching Scheme (In Hours)			Total Credits (L+T+P/2)	Examination Scheme				
				Theory Marks		Practical Marks		Total Marks
L	T	P	C	CA	ESE	CA	ESE	
3	-	4	5	30*	70	25	25	150

(*):Out of 30 marks under the theory CA, 10 marks are for assessment of the micro-project to facilitate integration of COs and the remaining 20 marks is the average of 2 tests to be taken during the semester for the assessing the attainment of the cognitive domain UOs required for the attainment of the COs.

Legends: ***L****-Lecture;* ***T*** *– Tutorial/Teacher Guided Theory Practice;* ***P*** *-Practical;* ***C*** *– Credit,* ***CA*** *- Continuous Assessment;* ***ESE*** *-End Semester Examination.*

5. SUGGESTED PRACTICAL EXERCISES

The following practical outcomes (PrOs) are the sub-components of the COs. *Some of the* ***PrOs*** *marked* ***'*'*** *are compulsory, as they are crucial for that particular CO at the 'Precision Level' of Dave's Taxonomy related to 'Psychomotor Domain'*.

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.	Build a program to count the frequency of words appearing in a string using a dictionary.	I	02
3.	Implement a Program for two matrix multiplication using simple nested loop and numpy module.	I	02
4.	Implement basic operations on arrays.	I	02
5.	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
6.	Design a class Complex for adding the two complex numbers and also show the use of constructor.	II	02
7.	Design a class for single level inheritance using public and private type derivation.	II	02
8.	Implement multiple and hierarchical inheritance.	II	02
9.	Write a Python program to demonstrate method overriding using inheritance.	II	02
10.	Implement push and pop algorithms of stack using list.	III	02
11.	Implement a program to convert infix notation to postfix notation using stack.	III	02
12.	Implement recursive functions.	III	02
13.	Implement a program to implement QUEUE using list that performs following operations: ENQUEUE, DEQUEUE, DISPLAY	III	04
14.	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.	IV	08

S. No.	Practical Outcomes (PrOs)	Unit No.	Approx. Hrs. required
	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		
15.	Implement a python program to search a particular element from list using Linear and Binary Search.	V	04
16.	Implement Bubble sort algorithm.	V	02
17.	Implement Selection sort and Insertion sort algorithm.	V	02
18.	Implement Merge sort algorithm.	V	02
19.	Implement construction of binary search trees.	VI	02
20.	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	08
			56 Hrs.

Note

*i. More **Practical Exercises** can be designed and offered by the respective course teacher to develop the industry relevant skills/outcomes to match the COs. The above table is only a suggestive list.*

*ii. The following are some **sample** 'Process' and 'Product' related skills(more may be added/deleted depending on the course)that occur in the above listed **Practical Exercises** of this course required which are embedded in the COs and ultimately the competency.*

S.No.	Sample Performance Indicators for the PrOs	Weightage in %
1	Identify suitable approach to implement logic	25
2	Correctness of data structure representation	20
3	Use python concepts to implement efficient program	25
4	Follow different input test cases to check output	10
5	Identify and mend coding errors in a program / Interpret the result and conclude	20
Total		**100**

6. MAJOR EQUIPMENT/ INSTRUMENTS REQUIRED

These major equipment with broad specifications for the PrOs is a guide to procure them by the administrators to usher in uniformity of practicals in all institutions across the state.

S. No.	Equipment Name with Broad Specifications	PrO. No.

S. 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	

7. AFFECTIVE DOMAIN OUTCOMES

The following ***sample*** Affective Domain Outcomes (ADOs) are embedded in many of the above mentioned COs and PrOs. More could be added to fulfill the development of this course competency.

a) Work as a leader/a team member.
b) Follow ethical practices.

The ADOs are best developed through the laboratory/field based exercises. Moreover, the level of achievement of the ADOs according to Krathwohl's 'Affective Domain Taxonomy' should gradually increase as planned below:

i. 'Valuing Level' in 1st year
ii. 'Organization Level' in 2nd year.
iii. 'Characterization Level' in 3rd year.

8. UNDERPINNING THEORY

The major underpinning theory is given below based on the higher level UOs of Revised Bloom's taxonomy that are formulated for development of the COs and competency. If required, more such UOs could be included by the course teacher to focus on attainment of COs and competency.

Unit	Unit Outcomes (UOs) (4 to 6 UOs at different levels)	Topics and Sub-topics
Unit – I **Basic Concepts of Data Structures**	1a. Define linear and non-linear data structures. 1b. Define time complexity and space complexity. 1c. Explain python specific - list, tuple, set, Dictionary and tuple data structures. 1d. Describe the Operations on arrays. 1e. Differentiate array and list.	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

Unit	Unit Outcomes (UOs) (4 to 6 UOs at different levels)	Topics and Sub-topics
		import numpy Operations on Arrays Arrays vs List
Unit – II **Basics of Object Oriented Programming**	2a. Explain concepts of Object Oriented programming. 2b. Explain the concept of class and object. 2c. Use a constructor to initialize an object. 2d. List the types of Inheritance. 2e. Use Inheritance to create re-usable codes in Python. 2f. Understand Polymorphism. 2g. Describe Abstract class.	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
Unit– III **Stack and Queues**	3a. Implement Stack Operations using List. 3b. List applications of Stack. 3c. Convert the given expression from Infix to Prefix/Postfix using stack. 3d. Evaluate the postfix expression using stack. 3e. Implement Queue Operations using List. 3f. Explain concepts of Circular queue. 3g. List applications of Queue. 3h. Differentiate circular and simple 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

Unit	Unit Outcomes (UOs) (4 to 6 UOs at different levels)	Topics and Sub-topics
Unit– IV **Linked List**	4a. Define a linked list. 4b. List types of Linked List. 4c. Implement basic operations on singly linked lists. 4d. Explain concepts of circular linked lists. 4e. Differentiate between circular linked list and singly linked list. 4f. Explain concepts of doubly linked lists. 4g. List applications of 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
Unit– V **Searching and Sorting**	5a. Design and Implement search algorithms. 5b. Arrange data in ascending and descending orders using appropriate sorting algorithms. 5c. Explain the working of the given sorting method step-by-step with an example and small data set.	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
Unit– VI **Trees**	6a. Describe a binary tree. 6b. Draw binary search tree for the given data set. 6c. Write algorithms to traverse the tree using the given method. 6d. List applications of trees.	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

9. SUGGESTED SPECIFICATION TABLE FOR QUESTIONPAPER DESIGN

Unit No.	Unit Title	Teaching Hours	Distribution of Theory Marks			
			R Level	U Level	A Level	Total Marks
I	Basic Concepts of Data Structures	04	04	02	00	06
II	Basics of Object Oriented Programming	08	04	04	04	12
III	Stack and Queues	08	02	06	06	14
IV	Linked List	08	04	08	02	14
V	Searching and Sorting	08	02	06	06	14
VI	Trees	06	02	04	04	10
Total		**42**	**18**	**30**	**22**	**70**

Legends: *R=Remember, U=Understand, A=Apply and above (Revised Bloom's taxonomy)*

Note*: This specification table provides general guidelines to assist students for their learning and to teachers to teach and question paper designers/setters to formulate test items/questions to assess the attainment of the UOs. The actual distribution of marks at different taxonomy levels (of R, U and A) in the question paper may slightly vary from above table.*

10. SUGGESTED STUDENT ACTIVITIES

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: Students should perform following activities in group and prepare reports of about 5 pages for each activity. They should also collect/record physical evidences for their (student's) portfolio which may be useful for their placement interviews:

a) Prepare a practical journal.
b) Undertake micro-projects in teams.
c) Give a seminar on any relevant topics.
d) Prepare a chart to classify Data structures.
e) Explore different python data structure modules.

11. SUGGESTED SPECIAL INSTRUCTIONAL STRATEGIES (if any)

These are sample strategies, which the teacher can use to accelerate the attainment of the various outcomes in this course:

a) Massive open online courses (***MOOCs***) may be used to teach various topics/subtopics.
b) Guide student(s) in undertaking micro-projects.
c) ***'L' in section No. 4*** means different types of teaching methods that are to be employed by teachers to develop the outcomes.
d) About ***20% of the topics/sub-topics*** which are relatively simpler or descriptive in nature is to be given to the students for ***self-learning***, but to be assessed using different assessment methods.
e) With respect to ***section No.10***, teachers need to ensure to create opportunities and provisions for ***co-curricular activities***.
f) Guide students for finding suitable data structures to solve given problems.

12. SUGGESTED MICRO-PROJECTS

Only one micro-project is planned to be undertaken by a student that needs to be assigned to him/her in the beginning of the semester. In the first four semesters, the micro-project

are group-based (group of 3 to 5). However, **in the fifth and sixth semesters**, the number of students in the group should **not exceed three**.

The micro-project could be industry application based, internet-based, workshop-based, laboratory-based or field-based. Each micro-project should encompass two or more COs which are in fact, an integration of PrOs, UOs and ADOs. Each student will have to maintain a dated work diary consisting of individual contributions in the project work and give a seminar presentation of it before submission. The duration of the micro project should be about **14-16 (fourteen to sixteen) student engagement hours** during the course. The students ought to submit micro-project by the end of the semester to develop the industry-oriented COs.

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

a) **Phone directory application using doubly-linked lists-** This project can demonstrate the working of contact book applications and also teach you about data structures like arrays, linked lists, stacks, and queues. Typically, phone book management encompasses searching, sorting, and deleting operations. A distinctive feature of the search queries here is that the user sees suggestions from the contact list after entering each character.
b) **Hangman Game:** The Hangman program randomly selects a secret word from a list of secret words. A random word (Eg. a fruit name) is picked up from our collection and the player gets limited chances to win the game. When a letter in that word is guessed correctly, that letter position in the word is made visible. In this way, all letters of the word are to be guessed before all the chances are over.
c) **Stack and queue implementation using linked list:** Develop a python program that implements stack and queue operations using linked list representation.

13. SUGGESTED LEARNING RESOURCES

S. 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

14. SOFTWARE/LEARNING WEBSITES

- Hands-On Data Structures and Algorithms with Python: Write complex and powerful code using the latest features of Python 3.7, 2nd Edition by Dr. Basant Agarwal, Benjamin Baka
- Data Structures and Algorithms with Python by Kent D. Lee and Steve Hubbard
- Problem Solving with Algorithms and Data Structures Using Python by Bradley N Miller and David L. Ranum
- https://nptel.ac.in/courses/106106145

15. PO-COMPETENCY-CO MAPPING

Semester III	Data Structures Using Python(Course Code: 4331601)						
	POs						
Competency & Course Outcomes	PO 1 Basic & Discipline specific knowledge	PO 2 Problem Analysis	PO 3 Design/ development of solutions	PO 4 Engineering Tools, Experimentation & Testing	PO 5 Engineering practices for society, sustainability & environment	PO 6 Project Management	PO 7 Life-long learning
Competency	Implement various types of data structures algorithms using python						
Course Outcomes CO a) Understand linear and non-linear data structures.	3	1	1	2	-	1	1
CO b) Implement Object Oriented Programming concepts in Python.	3	2	2	3	-	2	1
CO c) Implement basic data structures such as stacks, queues and linked lists.	3	2	2	3	-	2	1
CO d) Apply Algorithms for solving problems like searching and sorting of data.	3	2	2	3	-	2	1
CO e) Implement non linear data structures like trees.	3	2	2	3	-	2	1

*Legend: '**3**' for high, '**2**' for medium, '**1**' for low and '**-**' for no correlation of each CO with PO.*

16. COURSE CURRICULUM DEVELOPMENT COMMITTEE

GTU Resource Persons

S. No.	Name and Designation	Institute	Email
1	Prof. Nandu A. Fatak	HOD-I.T. LEC Poly Morbi	nandu_fatak@yahoo.com
2	Ms. Ayesha S. Shaikh	RCTI,Sola, Ahmedabad	shaikh.ayesha0014@gmail.com
3	Mr. Hardik Mandora	RCTI,Sola, Ahmedabad	hmandora@gmail.com
4	Mr. Pradipsinh K. Chavda	LECP Morbi	pradipchavda.it@gmail.com