LINGAYA'S VIDYAPEETH
choose to know

Lingaya's Vidyapeeth

Deemed-to-be-University u/s 3 of UGC Act 1956, Government of India
NAAC ACCREDITED
Approved by MHRD / AICTE / PCI / BCI / COA / NCTE
Nachauli, Jasana Road, Faridabad- 121002 (Haryana)
Website: www.lingayasvidyapeeth.edu.in | Ph: 0129-2598200-05

COURSE PLAN & COURSE DATA SHEET

PROGRAM: BCA	DEGREE: BCA
COURSE: Data Structure Using C	SEMESTER: 2nd CREDITS: 3
COURSE CODE: BCA-102 REGULATION:	COURSE TYPE: CORE
COURSE AREA/DOMAIN: Computer Applications	CONTACT HOURS: 42
CORRESPONDING LAB COURSE CODE (IF ANY): BCA-152	LAB COURSE NAME (IF ANY): Data Structure Using C LAB

PROGRAM EDUCATIONAL OBJECTIVES:

Program Educational Objectives for a course on data structures using C typically focus on the skills, knowledge, and attributes that students are expected to acquire after completing the program. Here are some potential PEOs for a Data Structures course using C:

1. Problem Solving Skills: Graduates will be able to analyze and solve complex problems using data structures and algorithms in the C programming language.
2. Algorithmic thinking: Graduates will develop proficiency in designing and implementing efficient algorithms to address a variety of computational problems.
3. Programming Proficiency in C: Graduates will demonstrate a strong command of the C programming language, specifically in the context of implementing and manipulating data structures.
4. Data Abstraction and Modularity: Graduates will be able to design and implement modular and reusable code using abstract data types, encapsulation, and information hiding.
5. Efficient Memory Management: Graduates will understand and apply efficient memory management techniques to optimize data storage and retrieval in various data structures.

SYLLABUS:

UNIT	DETAILS	HOURS
I	INTRODUCTION TO DATA STRUCTURES: Definition of data structure, data structure operations. Algorithms: Complexity, Time Space tradeoff, Complexity of Algorithms, Asymptotic Notations for Complexity of Algorithms, Variables.	7
II	ARRAYS AND LINKED LISTS: Introduction, Linear arrays, Representation of linear arrays in memory, Address calculation of using row and column major ordering, Traversing linear arrays, Inserting and Deleting, Multidimensional arrays, Linked Lists, Representation of Linear Lists in memory, Traversing a Linked List, Searching a linked List, Insertion into a linked list, Deletion from linked list, Circular linked lists, Doubly linked lists, Header linked lists, Memory allocation: Garbage collection, overflow and underflow.	8
III	STACK AND QUEUES: Stacks: Definition, Array representation of stacks, Linked representation of stacks, Polish notation, Evaluation of a Postfix Expression, Transforming Infix Expressions into Postfix Expressions, Queues: Definition, Array representation of Queues, Linked representation of Queues, Circular queues, Priority Queue, Double Ended Queue.	9
IV	TREES AND GRAPHS: Definition of trees and Binary trees; Properties of Binary trees and Implementation; Binary Traversal pre-order; post order; in- order traversal; Binary Search Trees, AVL trees, Balanced trees. Definition of Undirected and Directed Graphs; The Array based implementation of graphs; Adjacency matrix; path matrix implementation; The Linked List representation of graphs; Graph Traversal – Breadth first Traversal; Depth first Traversal.	11
V	SORTING AND SEARCHING ALGORITHMS: Introduction; Sorting by exchange; selection; insertions; bubble sort; Merge sort; Quick sort, Heap sort; Searching Algorithms: Straight Sequential Search; Binary Search (recursive & non–recursive Algorithms).	7

TOTAL HOURS	42

Teacher Centric Approach
TC1: Chalk and Talk, TC2: PPT, TC3: Video Lectures TC4: Blended learning
Learner Centric Approach:
LC1: Assignment. LC2: Mini project. LC3: Quiz/Class test. LC 4: Seminar on recent trends.
LC5: Group Task. LC6: Others

DETAILED SESSION PLAN

Lecture session/ Number	Topics to be covered	CO addressed	Teacher Centric Approach	Learner Centric Approach	References	Relevance with POs and PSOs
1	Definition of data Structures and abstract data types; linear vs. non-linear data structure		TC1, TC2	LC1,LC3	T1/T2/R1	1
2	Primitive vs. non-primitive data structure; static and dynamic implementations		TC1,TC2	LC1,LC3	T1/T2/T3/R1	1
3	Arrays, 1,2-dimensional arrays, insertion & deletion in 1-D array; examples and real life applications.		TC1,TC2	LC1,LC3	T1/T2/R1/R2	2
4	Time complexity; Big Oh notation		TC1,TC2	LC1,LC3,LC4	T1/T2/T3/R1/R2	2
5	Best case, worst case, average case; factors depends on running time		TC1,TC2	LC1,LC3	T1/T2/T3/R1/R2	2
6	Introduction to recursion. ABQ1		TC1,TC2	LC1,LC3	T1/T2/R1/R2/R3	1

7	QUIZ-1	TC1,TC2	LC1,LC3,LC4	T1/T2/T3/R1/R2	
8	Stacks: definition, array based	--	--	--	2
9	Applications of Stack-infix, postfix, prefix	TC1,TC2	LC1,LC3,LC4	T1/T2/T3/R1/R2	2
10	Infix, postfix, prefix Coversions	TC1,TC2	LC1,LC3	T1/T2/T3/R1/R2	2
11	Definition of queues, circular queue.	TC1,TC2	LC1,LC3,LC2	T1/T2/T3/R1/R3	1
12	Array based implementation of	TC1,TC2	LC1,LC3,LC2	T1/T2/R1/R3	2
13	ASSESSMENT-1	--	--	--	
14	LINKED LISTS introduction , different	TC1,TC2	LC1,LC3	T1/T2/R1/R3	1
15	Implementation of singly linked list,	TC1,TC2	LC1,LC3	T1/T2/T3/R1/R2	2
16	Linked list implementation of	TC1,TC2	LC1,LC3	T1/T2/T3/R1/R3	2
17	Implementation of circular linked list	TC1,TC,2	LC1,LC3	T1/T2/T3/R1/R3	2
18	Implementation of doubly linked list,	TC1,TC2	LC1,LC3	T1/T2/R1/R2	2
19	QUIZ-2	TC1,TC2	LC1,LC3	T1/T2/R1/R2	
20	TREES AND GRAPHS: Definition	--	--	--	1
21	Implementation of binary trees.	TC1, TC2	LC1,LC3	T1/T2/T3/R1/R3	2
22	Binary traversal pre-order, post-order, in-	TC1,TC2	LC1,LC3	T1/T2/T3/R1/R3	2
23	Introduction of binary search trees and	TC1,TC2	LC1,LC3, LC4	T1/T2/T3/R1/R2	2
24	Insertion & deletion operation of BST.	TC1,TC2	LC1,LC3	T1/T2/T3/R1/R2	2
25	Definition of undirected and directed	TC1,TC2	LC1,LC3	T1/T2/T3/R1/R2	2
24	Adjacency matrix; path matrix implementation	TC1,TC2	LC1,LC3	T1/T2/R1/R2	2
25	Linked list representation of	--	--	--	2
26	Graph traversal: breadth first traversal,	TC1,TC2	LC1,LC3	T1/T2/R1/R3	1
27	Implementations and applications.	TC1,TC2	LC1,LC3,LC5	T1/T2/R1/R3	1
28	ASSESSMENT-2	TC1,TC2	LC1,LC3,LC5	T1/T2/R1/R2	
29	Introduction to sorting and searching of	TC1,TC2	LC1,LC3,LC5	T1/T2/R1/R2	1
30	Selection, insertions, bubble sort	TC1,TC2	LC1,LC3,LC5	T1/T2/R1/R2	2

31	Merge sort, merging of sorted arrays and		TC1,TC2	LC1,LC3	T1/T2/R2/R3	2
32	Heap sort, searching algorithms: straight		TC1,TC2	LC1,LC3	T2/T3/R1/R2	2
33	Binary search (recursive &		--	--	T1/T2/T3/R1/R2	2
34	Revision Class		--	--	--	
35	Mathematical solutions for sorting algorithms		TC1, TC2	LC1,LC3	T1/T2/T3/R1/R3	2
36	Merging techniques		TC1,TC2	LC1,LC3	T1/T2/T3/R1/R3	2
37	Graph implementation and implications		TC1,TC2	LC1,LC3, LC4	T1/T2/T3/R1/R2	2
38	Significance of linked lists		TC1,TC2	LC1,LC3	T1/T2/T3/R1/R2	2
39	Difference between various linked lists		TC1,TC2	LC1,LC3	T1/T2/T3/R1/R2	2
40	Current trends and data structure implication		TC1,TC2	LC1,LC3	T1/T2/R1/R2	2
41	Quiz-3		--	--	--	
42	Final Assessment		--	--	--	

TEXT/REFERENCE BOOKS:

T/R	BOOK TITLE/AUTHORS/PUBLICATION
1	Langsam, Augentem M.J. and Tenenbaum A. M., Data Structures using C & C++‖, Prentice Hall of India, 2009.
2	R. S.Salariya, Data Structure and Algorithm, Khanna Publications.
3	Aho A. V., Hopcroft J. E. and Ullman T. D., ―Data Structures and Algorithms‖, Original Edition, Addison-Wesley, Low Priced Edition, 1983.

 # WEB SOURCE REFERENCES (W):

1	Geeksforgeeks
2	www.coursera.com
3	www.simplilearn.com

COURSE PRE-REQUISITES:

C.CODE	COURSE NAME	DESCRIPTION	SEM
-	Basic knowledge of computers	-	-

COURSE OBJECTIVES:

1. **Understand Basic Data Structures:** Develop a clear understanding of fundamental data structures such as arrays, linked lists, stacks, and queues. Learn the principles behind their implementation and comprehend the advantages and limitations of each structure.
2. **Implement and Manipulate Trees and Graphs:** Gain proficiency in implementing and manipulating tree structures (e.g., binary trees, AVL trees) and graph structures. Understand the algorithms for tree traversal, graph traversal, and basic operations on these structures.
3. **Apply Sorting and Searching Techniques:** Learn and implement various sorting algorithms (e.g., bubble sort, quicksort, merge sort) and searching algorithms (e.g., linear search, binary search). Understand the time and space complexity of these algorithms and their application in different scenarios.
4. **Develop Dynamic Data Structures:** Explore dynamic data structures like linked lists and dynamic arrays. Understand memory management techniques, such as pointers and dynamic memory allocation, and their role in creating flexible and efficient data structures.
5. **Analyze Algorithm Complexity:** Learn to analyze the time and space complexity of algorithms. Understand Big O notation and apply it to evaluate the efficiency of algorithms and data structures. Gain insights into choosing the most appropriate data structure based on the requirements of a particular problem.

COURSE OUTCOMES:

S.NO	DESCRIPTION	PO(1..12) MAPPING	PSO(1..3) MAPPING
CO1	Understand the concept of data structures, algorithms, time and space complexity.	PO1,PO2	PSO1
CO2	Understand basic data structures such as arrays and linked lists.	PO1,PO2,PO3	PSO1,PSO2
CO3	Describe the data structures such as stacks and queues.	PO1,PO2,PO3,PO4,PO5	PSO1,PSO2
CO4	Solve problems involving graphs and trees	PO1,PO2,PO3	PSO1,PSO2
CO5	Apply Algorithm for solving problems like sorting, searching, insertion and deletion of data.	PO1,PO2,PO3,PO4,PO5	PSO1,PSO2
COURSE OVERALL PO/PSO MAPPING:			

COURSE OUTCOMES VS POs MAPPING (DETAILED; HIGH:3; MEDIUM:2; LOW:1**):**

S.NO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	2	1	-	-	-	-	-	-	1	-	-	-	2	-	1
CO2	2	1	1	2	-	-	1	1	-	-	1	-	2	1	-
CO3	2	2	1	1	1	-	-	2	-	1	-	-	2	2	-
CO4	2	1	1	-	-	1	1	1	-	-	1	1	2	2	1
CO5	2	1	1	1	1	-	-	-	1	1	1	-	2	2	-

** For Entire Course, PO & PSO Mapping*

POs & PSO REFERENCE:

PO1	Engineering Knowledge	PO7	Environment & Sustainability	PSO1	To equip the students with theoretical and implementation knowledgebase in all the latest areas of Computer Science; Engineering for a successful career in software industries, pursuing higher studies, or entrepreneurial establishments.
PO2	Problem Analysis	PO8	Ethics	PSO2	To nurture the students with the critical thinking abilities for better decision making by offering them a socially acceptable solutions to real life problems through computing paradigm.

PO3	Design & Development	PO9	Individual & Team Work	PSO3	To nurture the students with the comprehensive analytical and design abilities by offering them techno-commercially feasible solutions of real business problems through computing.
PO4	Investigations	PO10	Communication Skills		
PO5	Modern Tools	PO11	Project Mgt. & Finance		
PO6	Engineer & Society	PO12	Life Long Learning		

COs VS POs MAPPING JUSTIFICATION:

S.NO	PO/PSO MAPPED	LEVEL OF MAPPING	JUSTIFICATION
Cxxx.1			
Cxxx.2			
Cxxx.3			
Cxxx.4			
Cxxx.5			
Cxxx*			

GAPS IN THE SYLLABUS - TO MEET INDUSTRY/PROFESSION REQUIREMENTS, POs & PSOs:

SNO	DESCRIPTION	PROPOSED ACTIONS
1	Advanced Sorting Algorithms: Introduce advanced sorting algorithms like merge sort, quicksort, and radix sort, and discuss their time and space complexities.	Need to be Covered in extra session
2	Hashing Techniques: Explore various hashing techniques such as open addressing, closed addressing, and perfect hashing. Discuss collision resolution strategies and hash function design.	Need to be Covered in extra session
3	Advanced Tree Structures: Cover advanced tree structures like AVL trees, Red-Black trees, B-trees, and Trie structures. Discuss their properties, applications, and advantages.	Need to be Covered in extra session
4	Graph Algorithms: Delve deeper into graph algorithms, including depth-first search (DFS), breadth-first search (BFS), Dijkstra's algorithm for shortest paths, and Kruskal's or Prim's algorithm for minimum spanning trees.	Need to be Covered in extra session
5	Dynamic Programming: Introduce the concept of dynamic programming and discuss how it can be applied to solve optimization problems efficiently. Explore examples such as the knapsack problem and longest common subsequence.	Need to be Covered in extra session

PROPOSED ACTIONS: TOPICS BEYOND SYLLABUS/ASSIGNMENT/INDUSTRY VISIT/GUEST LECTURER/NPTEL ETC

 # TOPICS BEYOND SYLLABUS/ADVANCED TOPICS/DESIGN:

1	Introduce advanced sorting algorithms like merge sort, quicksort, and radix sort, and discuss their time and space complexities.
2	Explore various hashing techniques such as open addressing, closed addressing, and perfect hashing. Discuss collision resolution strategies and hash function design.
3	Cover advanced tree structures like AVL trees, Red-Black trees, B-trees, and Trie structures. Discuss their properties, applications, and advantages.
4	Delve deeper into graph algorithms, including depth-first search (DFS), breadth-first search (BFS), Dijkstra's algorithm for shortest paths, and Kruskal's or Prim's algorithm for minimum spanning trees.
5	Introduce the concept of dynamic programming and discuss how it can be applied to solve optimization problems efficiently. Explore examples such as the knapsack problem and longest common subsequence.

DELIVERY/INSTRUCTIONAL METHODOLOGIES:

☐ CHALK & TALK	☐ STUD. ASSIGNMENT	☐ WEB RESOURCES	☐ NPTEL/OTHERS
☐ LCD/SMART BOARDS	☐ STUD. SEMINARS	☐ ADD-ON COURSES	☐ WEBNIARS

ASSESSMENT METHODOLOGIES-DIRECT

☐ ASSIGNMENTS	☐ STUD. SEMINARS	☐ TESTS/MODEL EXAMS	☐ UNIV. EXAMINATION
☐ STUD. LAB PRACTICES	☐ STUD. VIVA	☐ MINI/MAJOR PROJECTS	☐ CERTIFICATIONS
☐ ADD-ON COURSES	☐ OTHERS		

ASSESSMENT METHODOLOGIES-INDIRECT

☐ ASSESSMENT OF COURSE OUTCOMES (BY FEEDBACK, ONCE)	☐ STUDENT FEEDBACK ON FACULTY (TWICE)
☐ ASSESSMENT OF MINI/MAJOR PROJECTS BY EXT. EXPERTS	☐ OTHERS

 # INNOVATIONS IN TEACHING/LEARNING/EVALUATION PROCESSES:

1. **Technology Integration:** Embrace and integrate technology tools in the classroom to enhance the learning experience. This can include interactive whiteboards, educational apps, virtual reality, and online collaboration platforms. Utilizing technology allows for more dynamic and interactive lessons, catering to diverse learning styles.
2. **Personalized Learning Paths:** Implement personalized learning approaches that cater to individual student needs and pace of learning. Adaptive learning platforms and data analytics can help tailor educational content, assignments, and assessments based on the strengths and weaknesses of each student, promoting a more customized learning experience.
3. **Active Learning Strategies:** Move away from traditional lecture-based approaches and incorporate active learning strategies. This involves engaging students in hands-on activities, group discussions, problem-solving exercises, and real-world projects. Active learning fosters critical thinking, collaboration, and practical application of knowledge.
4. **Blended Learning Models:** Adopt blended learning models that combine face-to-face instruction with online resources. This allows for flexibility in learning, enabling students to access materials at their own pace outside the classroom. Flipped classrooms, where students learn new concepts online and engage in discussions and activities during class, are an example of a blended learning approach.
5. **Assessment Innovation:** Rethink assessment methods to go beyond traditional exams and quizzes. Explore alternative forms of assessment, such as project-based assessments, portfolios, presentations, and peer assessments. Additionally, incorporate formative assessments and feedback throughout the learning process to help students track their progress and make improvements.

Prepared by
Ms. Tanya Chauhan

Approved by
(HOD)