2017-18

Prabodh C P
Assistant Professor
CSE, SIT-Tumakuru
Mob: 8151894110

Course plan for ODD 2017-18

[SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMAKURU]

COURSE PLAN FOR THE ACADEMIC YEAR *ODD 2016-2017*

***Semester:* III**

Course Title: **Data Structures**	*Course Code*: **3CCI02**
Total Contact Hours: **52 + 13T Hrs**	*Duration of SEE*: **3 Hrs**
SEE Marks: **50**	*CIE Marks*: **50**
Lesson Plan Author : **PRABODH C P**	*Date*: 28-07-2017

Prerequisites:

Knowledge of fundamentals of C programming, problem-solving, design, and implementation skills and basic mathematics is essential.

Course Overview:

- The primary objective is to provide the student with an advanced treatment of computer programming with an emphasis on design and implementation of abstract data structures.
- Choose the data structures that effectively model the information in a problem.
- Judge efficiency trade-offs among alternative data structure implementations or combinations.
- To write Program effectively with pointers, arrays, structures, and dynamically allocated memory and describe their internal representations.
- To demonstrate understanding of the abstract properties of various data structures such as stacks, queues, lists, and trees.
- Comparisons of different implementations of data structures and to recognize the advantages and disadvantages of the different implementations.

Course Learning Objectives – CLOs

At the end of the course the student should be able to:

1. Describe the properties of various data structures such as stacks, queues, lists, and trees.
2. Implement the data structures such as stacks, queues, lists, and trees using C language.
3. Compare different implementations of Data Structures and to recognize the advantages and disadvantages of the different implementations
4. Explain the applications of various Data Structures.
5. Choose an appropriate Data Structure for a given problem.

Course Outcomes - COs

Upon successful completion of the course the student will be able to:

CO1: Apply advanced C programming techniques like pointers, dynamic memory allocation, structures to developing solutions for particular problems;

CO2: Design and implement abstract data types such as linked list, stack, queue and tree by using C as the programming language using static or dynamic implementations techniques.

CO3: Decide, Design and **Develop** a solution for a given problem by **Selecting** an appropriate data structure for a given open end problem.

CO4: Implement and submit comprehensive and continuous assignment modules on societal problems as a team event (PO: 11(L) and PO: 12(L))

CO5: Do a **Survey** on the complex implementation challenges related to the data structures in real world scenario.

Mapping of Course outcomes with Program outcomes:

Course outcomes	POs												PSOs		
	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3
CO1	**L**	**L**													
CO2		**L**	**L**										**M**		
CO3				**L**	**M**				**L**				**M**	**L**	**L**
CO4									**M**		**L**	**L**	**L**		**L**
CO5												**L**	**L**		

Degree of compliance **L**: Low **M**: Medium **H**: High

SCHEDULE PLAN FOR ACADEMIC YEAR 2017-18

* Please note that this schedule may change as we progress through the course material

Sl. No.	Date	Course Content	Teaching supplement	Student activity	Status	Deviation details
		UNIT - 1				
1.	31-JUL-2017	Introduction to Data Structures	Black Board			
2.	01-AUG-2017	Review of C programming Concepts	Black Board			
3.	02-AUG-2017	**Structures, unions and Dynamic memory allocation:** Defining the structures, Declaring structure variables, Accessing structure members,	Black Board			
4.	03-AUG-2017	Structure initialization, copying and comparing structure variables,	Black Board			
5.	04-AUG-2017	operations on individual members,	Black Board			
6.	07-AUG-2017	Array of structures, Programs.	Black Board			
7.	08-AUG-2017	unions, size of structures, bit fields,	Black Board			
8.	09-AUG-2017	Dynamic memory allocation functions.	Black Board			
9.	10-AUG-2017	**File management in C:** Defining and opening a file, closing a file,	Black Board			
10.	14-AUG-2017	input/output operations on files,	Black Board			
11.	16-AUG-2017	Programs on files.	Black Board			
12.	17-AUG-2017	Error handling during I/O operations,	Black Board			

13.	21-AUG-2017	random access to files,	Black Board			
14.	22-AUG-2017	Command line arguments.	Black Board	Group Activity: Implementing a System Command		
UNIT - 2						
15.	23-AUG-2017	**The Stack:** Definition and Examples, representing Stacks in C	Black Board			
16.	24-AUG-2017	Example: Infix, Postfix, and Prefix.	Black Board			
17.	28-AUG-2017	**Recursion:** Recursive Definition and Processes, Recursion in C,	Black Board			
18.	29-AUG-2017	Writing recursive programs: The Towers of Hanoi Problem, Efficiency of Recursion.	Black Board	Assignment-I		
19.	30-AUG-2017	Programs on Stack	Black Board			
20.	31-AUG-2017	Programs on Stack	Black Board			
	04-SEPT-2017	**Queues and Lists:** The Queue and Its Sequential Representation: C implementation of Queues,	Black Board			
21.	05-SEPT-2017	Insertion, Deletion and Display operations,	Black Board			
22.	06-SEPT-2017	Types of Queues (Linear, Circular, Priority and Double Ended Queues).	Black Board			
23.	07-SEPT-2017	Programs on Queues	Black Board	Group Activity: Implementing a Process Scheduler		
24.						
UNIT – 3						
25.	11-SEPT-2017	**Queues and Lists Continued** Linked lists : Inserting and removing nodes from a list ,	Black Board			
26.	12-SEPT-2017	Linked implementation of stacks, getnode and freenode operations,	Black Board			
27.	13-SEPT-2017	linked implementation of queues,	Black Board			
28.	14-SEPT-2017	Examples of list operation,	Black Board			
29.	25-SEPT-2017	List implementation of priority queues,	Black Board			
30.	26-SEPT-2017	Header nodes.	Black Board	**First Test**		
31.	27-SEPT-2017	**Lists in C:** Allocating and freeing dynamic variables,	Black Board			
32.	28-SEPT-2017	Programs on lists	Black Board			
33.	03-OCT-2017	Linked lists using dynamic variables,	Black Board			
34.	04-OCT-2017	Queues as lists in C,	Black Board			
35.	09-OCT-2017	Examples of list operations in C,	Black Board			
36.	10-OCT-2017	Non- integer and non homogeneous lists,	Black Board	Assignment-II		
37.	11-OCT-2017	Comparing dynamic and array implementations of lists,	Black Board			

38.	12-OCT-2017	Implementing header nodes.	Black Board	Group Activity : Implementing a Phone Contact Manager		
UNIT - 4						
39.	16-OCT-2017	**Other List Structures:** Circular lists,	Black Board			
40.	17-OCT-2017	Stack as a circular list,	Black Board			
41.	19-OCT-2017	Queue as a circular list,	Black Board			
42.	23-OCT-2017	Primitive operations on circular lists,	Black Board			
43.	24-OCT-2017	The Josephus problem,	Black Board			
44.	25-OCT-2017	Header nodes,	Black Board	**Second Test**		
45.	26-OCT-2017	Programs on circular lists	Black Board			
46.	30-OCT-2017	Doubly linked lists,	Black Board			
47.	31-OCT-2017	Primitive operations on Doubly linked list.	Black Board			
UNIT - 5						
48.	07-Nov-2017	**Trees:** Operations on Binary Trees,	PPT Slides			
49.	08-Nov-2017	Applications of Binary Trees	PPT Slides			
50.	09-Nov-2017	Binary Tree Representations: Node representation of Binary Trees,	PPT Slides			
51.	16-Nov-2017	Internal and External Nodes, Implicit array representation of Binary Trees,	PPT Slides			
52.	18-Nov-2017	Binary tree traversals: inorder, preorder, postorder,	PPT Slides			
53	20-Nov-2017	Threaded Binary Trees	PPT Slides			
54	21-Nov-2017	Trees and Their applications: C Representations of Trees,	PPT Slides			
55	22-Nov-2017	Tree Traversals, General Expressions as Trees,	PPT Slides			
56	23-Nov-2017	Evaluating an Expression Tree, Constructing a Tree.	PPT Slides			

Tutorial class details:

Sl. No	Date	Topics to be Covered
1.	11-Aug-2017	Review of C programming
2.	18-Aug-2017	Review of C programming
3.	1-Sep-2017	Programs on Stacks
4.	8-Sep-2017	Programs on Stacks

5.	15-Sep-2017	Programs on Queues
6.	22-Sep-2017	Programs on Queues
7.	29-Sep-2017	Programs on Singly Linked Lists
8.	6-Oct-2017	Programs on Singly Linked Lists
9.	13-Oct-2017	Programs on Doubly Linked Lists
10.	26-Oct-2017	Programs on Circular Linked Lists
11.	27-Oct-2017	Programs on Trees
12.	10-Oct-2017	Programs on Trees
13.	18-Nov-2017	Programs on Trees
14.	24-Nov-2017	Programs on Trees

TEXT BOOKS:

1	E. Balaguruswamy	**Programming in ANSI C,** 4th Edition, Tata McGraw-Hill Publications
2	Yedidyah Langsam, MosheJ. Augenstein, AaronM. Tenenbaum	Data structures using C and C++,PHI/Pearson, 2nd Edition . (Chapter 2, 3.1, 3.2, 3.3(only the Towers of Hanoi Problem), 3.5. 4.1(excluding Queue as an ADT), 4.2, 4.3(except array implementation of list, Limitations of array implementation, comparing dynamic and array implementations of list), 4.5(except addition of long positive integers using circular and doubly linked list), 5.1, 5.2(except choosing Binary Tree Representation, Traversal using a Father field, Heterogeneous Binary Trees), and 5.5)

REFERENCE BOOK:

1	Jean-Paul Tremblay, Paul G. Sorenson	An Introduction To Data Structures With Applications, 2nd edition, McGraw-Hill International Editions

Assessment:

Course outcome	Method of assessment	Assessed during
CO1	Based on marks obtained.	Quiz , Assignment, Tests, SEE
CO2	Marks obtained	
CO3	Based on marks obtained	
CO4	Based on marks obtained in implementation and the design report	CIE-Continuous and Comprehensive Assignment module
CO5	Based on marks obtained	CIE-Survey Report

Evaluation Scheme : CIE Scheme

Assessment	Weightage of Marks
Quiz – 1	03
Quiz – 2	03
Test – 1	17
Quiz – 3	03
Test – 2	17
Quiz – 4	03
Assignments	4
Total	**50**

Course Unitization for I, II Test and Semester End Examination:

Modules	Topics		Teachin Hours	No. of Questions in		No. of Questions in SEE
				I Mid sem	II Mid sem	
1	1	Structures and Unions, Array of structures, Unions, size of structures, bit fields.	6	1/2	--	1
	2	File management: Defining & opening a file,	4	1	--	
2	3	Linear Data Structures and Sequential Storage Representation, The Stacks:	4	1	--	1
	4	Recursion, The Queues	6	1	--	
3	5	Linear Data Structures and their linked Storage Representation	10	1/2	1/2	1
4	6	Linear Data Structures – Doubly and Circular linked lists and Applications	10	--	1/2	1
5	7	Non- Linear Data Structures , Trees, Binary search tree	12	--	--	1
				All questions are compulsory	All questions are compulsory	First question is compulsory and answer any four questions from questions **two** to **six**.

UNIT I

MODULE WISE PLAN

*Course Code & Title***: 3CCI02, Data Structures**	
Chapter Title: **Structures and Unions, Array of structures, Unions, bit fields. File management: Defining & opening a file**	*Planned Hours:***10L+02T Hrs**

Learning Objectives

At the end of the chapter the student should be able to:

1. **Explain** a mechanism of packing data of different types of variables under a single name.(L2)
2. **Draw** the logical structure of the members of the structure.(L2)
3. **Differentiate** between an array and structures.(L3)
4. **Implementation** of the unions and bit fields.(L4)
5. **Differentiate** between the unions and structures**.**(L2)
6. **Define** a file, what are the different operations performed with file and their syntaxes (L2).
7. **Design a** new structure for a given application problem.(L3)
8. **Explain** the concepts of bit fields and its advantages. (L3).
9. **Write** programs various operations performed with files.(L3)
10. **Write** program by using command line arguments.(L3)

Model Questions

1. What is a data structure?(L1)
2. Explain the different ways of declaration and initialization of structures. (L2)
3. Write a program to implement the structure which consists of student information and print the data on the console. (L2)
4. Differentiate between (L2)
   a. Arrays and structures
   b. Unions and structures
5. Write a program to read the data from the given file and write the read contents into another file. (L1)
6. Write a program to read the numbers from the file and write a program to separate the no's as odd and even no's into two separate files. (L1)

UNIT 2

MODULE WISE PLAN

Course Code & Title: **3CCI02, Data Structures**	
*Chapter Title: **The Stack,Recursion,Queues***	*Planned Hours:***11L+03T Hrs**

Learning Objectives

At the end of the chapter the student should be able to:

1. **Distinguish between** stacks and queues.(L2)
2. **Describe** the structure of a Stack and its applications.(L3)
3. **Distinguish between** recursive and non-recursive algorithms. (L3)
4. **Explain** primitive operations that can performed on stack.(L2)
5. **Implementation** of stacks using structures. (L1)
6. **List** various types of queues.(L1)
7. **Distinguish between** ordinary queue and priority queue.(L3)
8. **Explain** primitive operations that can performed on queues.(L2)
9. **Write** a recursive functions for various problems. (L4)
10. **Distinguish between** dequeue and circular queue.(L3)

Model Questions

1. Differentiate between stack and queue.(L2)
2. Write C functions and explain the PUSH and POP operations.(L4)
3. Write a C program to implement queue operations using structures.(L4)
4. Explain the different types of queues and its applications.(L2)
5. What is recursion? Write a recursive function to compare nth term of a Fibonacci series. Give the trace along with stack contents for n=4.(L3)
6. Write C functions and explain the INSERT, DELETE and DISPLAY operations with the circular queue.(L4)

UNIT 3

MODULE WISE PLAN

Course Code & Title: **3CCI02, Data Structures**	
Chapter Title: **Linked lists,Lists in C: Linear Data Structures and their linked Storage Representation**	*Planned Hours:* **12L+03T Hrs**

Learning Objectives

At the end of the chapter the student should be able to:

1. **Explain** the need of the Dynamic memory allocation.(L3)
2. **Explain** the different types of lists and their storage representation.(L1)
3. **Implementation** of the stacks and queues using dynamic memory allocation.(L2)
4. **Distinguish** between the array implementation and linked list representations.(L3)
5. **Write** the list examples using c to store both integer and non-integer form.(L4)
6. **Define** a list and the basic operations performed on it.(L1)
7. **Differentiate** between the static memory allocation and dynamic memory allocation.(L2)
8. **Differentiate** between the calloc()and malloc dynamic memory allocation funcitons. (L2)
9. **Write** programs to implement the singly linked list with the header node.(L4)
10. **Write** programs to implement primitive operations with the singly linked list.(L4)

Model Questions

1. Give the syntax of different types of dynamic memory allocation functions in C.(L3)
2. Explain how linked list representation of stacks is more efficient than array representation.(L1)
3. Write a C program to count the no. of nodes in a singly linked list.(L1).
4. Distinguish between the malloc() and calloc() functions.(L3)
5. Write a C program to delete a node form the front end and insert it at the end of the list.(L3)
6. Write a program to delete a node from a singly linked list whose position is specified.(L3).
7. Distinguish between the array implementation and linked list representations.(L3)
8. Write a program to delete all occurrences of a given node from a singly linked list.(L3).

UNIT 4

MODULE WISE PLAN

Course Code & Title: : 3**CCI02, Data Structures**	
Chapter Title: **Other List Structures: Linear Data Structures – Doubly and Circular linked lists and Applications**	*Planned Hours:* **09L+02T Hrs**

Learning Objectives

At the end of the chapter the student should be able to:

1. **Define** doubly linked list.(L1)
2. **Write** programs to implement the primitive operations with doubly linked list.(L6)
3. **Differentiate** between circular linked lists and doubly linked list.(L2)
4. **List** different types of lists.(L1)
5. **Explain** the purpose of using the circular lists and doubly linked lists.(L2)
6. **Describe** Josephus problem.(L2)
7. **Differentiate** between circular singly linked list and circular doubly linked list (L4)
8. **Describe** the usage of header nodes with lists.(L2)
9. **Define** circular doubly linked list.(L1)
10. **Write** programs to add two integers using doubly linked list.(L6)

Model Questions

1. Define a circular linked list with header.(L1)
2. Explain the Josephus problem with example.(L2)
3. Write a program to add/subtract two polynomials.(L4)
4. Write functions to perform the following.
    a. To find the sum of all the elements in a list.
    b. To insert a new node in the specified position in the list.
5. How long positive numbers can be represented using circular singly linked lists? Explain with an example.
6. Write a program to implement the following functions with the circular doubly linked lists with the header node.
    a. To insert a new node in the beginning
    b. To insert a node X in the position Y.
    c. To display contents of the list in reverse order.

UNIT 5

MODULE WISE PLAN

Course Code & Title: : **3CCI02, Data Structures**	
Chapter Number & Title: **Non- Linear Data Structures: Trees, Binary Search Trees, Threaded binary trees**	*Planned Hours:* **10L+03T Hrs**

Learning Objectives

At the end of the chapter the student should be able to:

1. **Define** the following (L1)
    a. Binary tree.
    b. All most complete binary tree
2. **Explain** the concept of representation of both array and linked list representation of trees. (L2)
3. **Write** program for creation of binary search tree. (L4)
4. **Describe** the benefits of using the trees and tree traversal techniques.(L3)
5. **Design** the heap trees and expression trees.(L3)
6. **Explain** the in-threaded binary tree with examples.(L2)

Model Questions

1. What is a binary tree? (L1)
2. Define the following terms: (L1)
    a. Forest
    b. Depth of the tree
    c. Ancestors of a node
3. Write C functions to perform the following operations.(L4)
    a. Creation of binary tree
    b. Insertion of a new node
    c. Deleting a new node
4. Write a program to evaluate the expression tree.(L4)
5. Explain different types of threaded binary trees with an example for each type. (L2)
6. Define a heap? Explain max heap and min heap with examples.(L3)

Programme Outcomes(POs)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solution in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PSOs:

PSO1: Professional Skills: The ability to understand, analyze, design and develop computer based systems for the benefit of society using the knowledge of Mathematics, Data Structures, Algorithms, Computer Architecture, System Software, Data Bases, Cloud Computing, Software Engineering, Web design, Big Data Analytics, Networking, Microcontrollers and other Computer Science related courses .

PSO2: Software development: The ability to understand the Engineering Principles and Methodologies of Software Systems and acquire professional skills of software development.

PSO3: Successful Career: The ability to practice modern tools, technologies and innovations in creating successful careers in academia and industry.

Faculty **Head of the Department** **Principal**