ALVA'S INSTITUTE OF ENGINEERING & TECHNOLOGY
(A Unit of Alva's Education Foundation)
Shobhavana Campus, Mijar-574225, Moodbidri , D.K - 574225
Phone: 08258-262725, Fax: 08258-262726
Affiliated to VTU Belagavi and Approved by AICTE, New Delhi, Recognized by Govt. of Karnataka

CALENDAR OF EVENTS (ODD SEMESTER 2022-23) BE & MBA

VISION

"Transformative education by pursuing excellence in Engineering and Management through enhancing skills to meet the evolving needs of the community"

MISSION

- To bestow quality technical education to imbibe knowledge, creativity and ethos to students community.
- To inculcate the best engineering practices through transformative education.
- To develop a knowledgeable individual for a dynamic industrial scenario.
- To inculcate research, entrepreneurial skills and human values in order to cater the needs of the society.

Week	Month	Days							Activities
		Mon	Tue	Wed	Thu	Fri	Sat	Sun	
1	SEP				1	2	3	4	19th : Commencement of VII Semester BE 22-23 : Student Mentoring 26 – 30 : Technical Talk/Club and Social Activity
2		5	6	7	8	9	10	11	
3		12	13	14	15	16	17	18	
4		19	20	21	22	23	24	25	
		26	27	28	29	30			
6	OCT						1	2	4th : Maha Navami & 5th : Vijaya Dashami 10th : Commencement of V Semester BE 20-21 : Student Mentoring 24th : Naraka Chaturdashi 24 – 29 : Technical Talk/Club and Social Activity 26th : Deepavali 31st : Commencement of III Semester BE
7		3	4	5	6	7	8	9	
8		10	11	12	13	14	15	16	
9		17	18	19	20	21	22	23	
10		24	25	26	27	28	29	30	
11		31							
12	NOV		1	2	3	4	5	6	1st : Kannada Rajyothsava 10th -12th : 1nd IA for VII Semester BE 24-25 – Student Mentoring 20– 25 Technical Talk/Club / Social Activity 28th : Commencement of 3rd Sem MBA
13		7	8	9	10	11	12	13	
14		14	15	16	17	18	19	20	
15		21	22	23	24	25	26	27	
16		28	29	30					
17	DEC				1	2	3	4	8 - 10 : 2nd IA for VII Semester / 1st IA V Semester BE 15 - 17 : 1st IA for BE III Semester BE 22-23 : Student Mentoring 29 - 31 : 3nd IA for VII Semester BE 26 - 31 : Technical Talk/Club / Social Activity **30-31 Second International Conference on Data Analytics & Learning-2022 (DAL'22)** 31 : Last Working Day of VII Semester BE
18		5	6	7	8	9	10	11	
19		12	13	14	15	16	17	18	
20		19	20	21	22	23	24	25	
21		26	27	28	29	30	31		
22	JAN-2023							1	2 - 4 : 2nd IA for V Semester BE / 1st IA for MBA 3rd Sem 14- Makara Sankranti 16-18 : 2nd IA for III Semester BE 20-21 : Student Mentoring 23 – 26 : Technical Talk/Club / Social Activity 24th,25th and 27th : 3nd IA for V Sem BE 26-Republic Day 27 : Last Working Day of V Semester BE
23		2	3	4	5	6	7	8	
24		9	10	11	12	13	14	15	
25		16	17	18	19	20	21	22	
26		23	24	25	26	27	28	29	
27		30	31						
28	FEB-2023			1	2	3	4	5	8 - 10 : 3nd IA for III Semester BE / 2nd IA for MBA 3rd sem 18- Maha Shivaratri 11 : Last Working Day of III Semester BE
29		6	7	8	9	10	11	12	
30		13	14	15	16	17	18	19	
31		20	21	22	23	24	25	26	
32		27	28						
33	MAR-2023			1	2	3	4	5	9-11 : 3nd IA for MBA 3rd sem 18 : Last Working Day of MBA 3rd Sem
34		6	7	8	9	10	11	12	
35		13	14	15	16	17	18	19	

Approved by IQAC Chairman

ALVA'S INSTITUTE OF ENGINEERING & TECHNOLOGY

Shobhavana Campus, Mijar, Moodbidri, D.K – 574225

Phone: 08258-262725, Fax: 08258-262726

Academic Time Table W.E.F. 28/11/2022

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

Academic Year	Scheme	Semester	Section	Room No	Class Coordinator
2022-23	2021	III	A	212	Mr. Venugopala Rao

Time / Day	9.00 To 9.50	9.50 To 10.40	10.40 To 11.00	11.00 To 11.50	11.50 To 12.40	12.40 To 1.40	1.40 To 2.30	2.30 To 3.20	3.30 To 5.00
MON	DS LAB (B1 BATCH)[AC + SO] / ADE LAB (B2 BATCH)		TEA BREAK		ADE	LUNCH BREAK	DSA	M-III	MMM
TUE	DS LAB (B2 BATCH) [AC + SO] / ADE LAB (B1 BATCH				Kannada		DSA	PT	M-III
WED	Mastering Office			COA	ADE		M-III	COA	PT
THU	DSA	COA		OOPJL (SBC)			COA	M-III	OOPJL (SNG)
FRI	OOPJL (SNG)	DSA		ADE	SCR		PT	M-III	MMM
SAT	COA	ADE		DSA	ADE				

Allocation of Subjects

Subjects			Staffs	Code
21MAT31	M-III	Transform Calculus, Fourier Series and Numerical Techniques - BSC	Ms. Sowmya	SM
21CS32	DSA	Data Structures and its Applications - IPCC	Ms. Soundarya B C Mr. Apurba Chakraboty Mr. Subramanya V Odeyar	SBC AC SO
21CS33	ADE	Analog and Digital Electronics - IPCC	Prof. Venugopal Rao	VR
21CS34	COA	Computer Organization and Architecture - PCC	Mr. Apurba Chakraborty	AC
21CSL35	OOPJL	Object Oriented Programming with JAVA Laboratory - PCC	Mr. Shrikanth N G Ms. Soundarya B C	SNG SBC
21UH36	SCR	Social Connect and Responsibility - UHV	Dr. Sapna	DS
21KSK37 /21KAK37	CIPE	Kannada	Dr. Yogish Kairody	YK
21CSL381	MO	Mastering Office [AEC}	Mr. Subramanya V Odeyar	SO
21NS83 21PE83 21YO83	NCMC	National Service Scheme (NSS) / Physical Education (PE) / Yoga	Prof. Venugopal Rao	VR
	PT	Placement Training	Prof. Harish Kunder	HK

BSC: Basic Science Course, **IPCC**: Integrated Professional Core Course, **PCC:** Professional Core Course, **HSMC:** Humanity and Social Science & Management Courses, **AEC**–Ability Enhancement Courses. **UHV:** Universal Human Value Course. **NCMC** - Non–credit mandatory courses

Time Table Coordinator **Head of the Department** **Principal**

ALVA'S INSTITUTE OF ENGINEERING & TECHNOLOGY

Shobhavana Campus, Mijar, Moodbidri, D.K – 574225

Phone: 08258-262725, Fax: 08258-262726

Academic Time Table W.E.F. 28/11/2022

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

Academic Year	Scheme	Semester	Section	Room No	Class Coordinator
2022-23	2021	III	A	212	Mr. Venugopala Rao

Time / Day	9.00 To 9.50	9.50 To 10.40	10.40 To 11.00	11.00 To 11.50	11.50 To 12.40	12.40 To 1.40	1.40 To 2.30	2.30 To 3.20	3.30 To 5.00
MON	DS LAB (B1 BATCH)[AC + SO] / ADE LAB (B2 BATCH)				ADE	LUNCH BREAK	DSA	M-III	MMM
TUE	DS LAB (B2 BATCH) [AC + SO] / ADE LAB (B1 BATCH				Kannada		DSA	PT	M-III
WED	Mastering Office		TEA BREAK	COA	ADE		M-III	COA	PT
THU	DSA	COA		OOPJL (SBC)			COA	M-III	OOPJL (SNG)
FRI	OOPJL (SNG)	DSA		ADE	SCR		PT	M-III	MMM
SAT	COA	ADE		DSA	ADE				

Allocation of Subjects

Subjects			Staffs	Code
21MAT31	M-III	Transform Calculus, Fourier Series and Numerical Techniques - BSC	Ms. Sowmya	SM
21CS32	DSA	Data Structures and its Applications - IPCC	Ms. Soundarya B C Mr. Apurba Chakraboty Mr. Subramanya V Odeyar	SBC AC SO
21CS33	ADE	Analog and Digital Electronics - IPCC	Prof. Venugopal Rao	VR
21CS34	COA	Computer Organization and Architecture - PCC	Mr. Apurba Chakraborty	AC
21CSL35	OOPJL	Object Oriented Programming with JAVA Laboratory - PCC	Mr. Shrikanth N G Ms. Soundarya B C	SNG SBC
21UH36	SCR	Social Connect and Responsibility - UHV	Dr. Sapna	DS
21KSK37 /21KAK37	CIPE	Kannada	Dr. Yogish Kairody	YK
21CSL381	MO	Mastering Office [AEC}	Mr. Subramanya V Odeyar	SO
21NS83 21PE83 21YO83	NCMC	National Service Scheme (NSS) / Physical Education (PE) / Yoga	Prof. Venugopal Rao	VR
	PT	Placement Training	Prof. Harish Kunder	HK

BSC: Basic Science Course, **IPCC**: Integrated Professional Core Course, **PCC:** Professional Core Course, **HSMC:** Humanity and Social Science & Management Courses, **AEC**–Ability Enhancement Courses. **UHV:** Universal Human Value Course. **NCMC** - Non–credit mandatory courses

Time Table Coordinator **Head of the Department** **Principal**

ALVA'S INSTITUTE OF ENGINEERING & TECHNOLOGY

Shobhavana Campus, Mijar, Moodbidri, D.K – 574225

Phone: 08258-262725, Fax: 08258-262726

Individual Faculty Time Table with effect from 28/11/2022

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

Academic Year	2022-23	Faculty Name	Ms. Soundarya B C (SBC)
Semester	ODD	Designation	Assistant Professor

Time / Day	9.00 To 9.50	9.50 To 10.40	10.40 To 11.00	11.00 To 11.50	11.50 To 12.40	12.40 To 1.40	1.40 To 2.30	2.30 To 3.20	3.30 To 5.00
MON	DSA		TEA BREAK			LUNCH BREAK	DSA (CSD)		
TUE	DSA						DSA (CSD)		DSA
WED	DSA				DSA		OOP with JAVA (AIML)		
THU	DSA (CSD)			OOP with JAVA (CSD)			DS LAB (B1 BATCH)		
FRI		DSA (CSD)					DS LAB (B2 BATCH)		
SAT				DSA (CSD)					

UNITS:	Theory:20	LAB: 10	Others: 02	TOTAL UNITS: 32

Allocation of Subjects (Subjects with Subject Code)

21CS32	Data Structures Laboratory [IPCC]
21CS32	Data Structures Laboratory [IPCC] - Dept of CSD
21CSL35	Object Oriented Programming with JAVA Laboratory - PCC
	Mentoring (2Hrs / Week)

Responsibilities

Class Coordinator,

Magazine, Website, E-News Letter Coordinator

Non- Credit activity, Internship, Review Paper, ERP Coordinator

Head of the Department
Dept. of Artificial Intelligence & Machine Learning
Alva's Institute of Engineering and Technology
Shobhavana Campus, Mijar
Moodubidire 574 225, D.K. Karnataka, India

Head of the Department

PRINCIPAL
Alva's Institute of Engg. & Technology,
Mijar, MOODBIDRI - 574 225

VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI

B.E. in Information Science and Engineering

Scheme of Teaching and Examinations2021

Outcome Based Education (OBE) and Choice Based Credit System (CBCS)

(Effective from the academic year 2021 - 22)

III SEMESTER

Sl. No	Course and Course Code	Course Title	Teaching Department (TD) and Question Paper Setting Board (PSB)	Teaching Hours /Week				Examination			
				Theory Lecture	Tutorial	Practical/ Drawing	Self -Study	Duration in hours	CIE Marks	SEE Marks	Total Marks
				L	T	P	S				
1	BSC 21MAT31	Transform Calculus, Fourier Series and Numerical Techniques	Maths	3	0	0		03	50	50	10
2	IPCC 21CS32	Data Structures and Applications	Any CS Board Department	3	0	2		03	50	50	10
3	IPCC 21CS33	Analog and Digital Electronics		3	0	2		03	50	50	10
4	PCC 21CS34	Computer Organization and Architecture		3	0	0		03	50	50	10
5	PCC 21CSL35	Object Oriented Programming with JAVA Laboratory		0	0	2		03	50	50	10
6	UHV 21UH36	Social Connect and Responsibility	Any Department	0	0	1		01	50	50	10
7	HSMC 21KSK37/47	Samskrutika Kannada	TD and PSB: HSMC	1	0	0		01	50	50	10
	HSMC 21KBK37/47	Balake Kannada									
	OR										
	HSMC 21CIP37/47	Constitution of India and Professional Ethics									
8	AEC 21CS38X/21 CSL38X	Ability Enhancement Course - III	TD: Concerned department PSB: Concerned Board	If offered as Theory Course: 1	0	0		01	50	50	10
				If offered as lab. course: 0	0	2		02			
								Total	400	400	8
9	Scheduled activities for III to VIII semesters	NMDC 21NS83	National Service Scheme (NSS)	NSS	All students have to register for any one of the cours National Service Scheme, Physical Education (PE)(Sp Athletics) and Yoga with the concerned coordinator of t during the first week of III semester. The activities shall b out from (for 5 semesters) between III semester to VIII SEE in the above courses shall be conducted during VIII examinations and the accumulated CIE marks shall be add SEE marks. Successful completion of the registered mandatory for the award of the degree. The events shall be appropriately scheduled by the college same shall be reflected in the colander prepared for the N Yoga activities.						
		NMDC 21PE83	Physical Education (PE) (Sports and Athletics)	PE							
		NMDC 21YO83	Yoga	Yoga							

Course prescribed to lateral entry Diploma holders admitted to III semester B.E./B.Tech programs

1	NCMC 21MATDIP31	Additional Mathematics - I	Maths	02	02	--	--	---	100	---	1

Note: BSC: Basic Science Course, **IPCC:** Integrated Professional Core Course, **PCC:** Professional Core Course, **INT** –Internship, **HSMC:** Hum Social Science & Management Courses, **AEC**–Ability Enhancement Courses. **UHV:** Universal Human Value Course.

L –Lecture, **T** – Tutorial, P- Practical/ Drawing, **S** – Self Study Component, **CIE:** Continuous Internal Evaluation, **SEE:** Semester End Examin Teaching Department, **PSB**: Paper Setting department

21KSK37/47 Samskrutika Kannada is for students who speak, read and write Kannada and **21KBK37/47** Balake Kannada is for non-Kannada reading, and writing students.

Integrated Professional Core Course (IPCC): Refers to Professional Theory Core Course Integrated with Practical's of the same course. Cred can be 04 and its Teaching–Learning hours (L : T : P) can be considered as (3 : 0 : 2) or (2 : 2 : 2). The theory part of the IPCC shall be evalu by CIE and SEE. The practical part shall be evaluated by only CIE (no SEE). However, questions from the practical part of IPCC shall be inclu SEE question paper. For more details, the regulation governing the Degree of Bachelor of Engineering /Technology (BE/B.Tech.) 2021-2 referred.

6. At the end of the 13th week of the semester

The sum of three tests, two assignments, and quiz/seminar/group discussion will be out of 100 marks and will be **scaled down to 50 marks**

(to have less stressed CIE, the portion of the syllabus should not be common /repeated for any of the methods of the CIE. Each method of CIE should have a different syllabus portion of the course).

CIE methods /question paper has to be designed to attain the different levels of Bloom's taxonomy as per the outcome defined for the course.

Semester End Examination:

Theory SEE will be conducted by University as per the scheduled timetable, with common question papers for the subject (**duration 03 hours**)

1. The question paper will have ten questions. Each question is set for 20 marks. Marks scored shall be proportionally reduced to 50 marks
2. There will be 2 questions from each module. Each of the two questions under a module (with a maximum of 3 sub-questions), **should have a mix of topics** under that module.

The students have to answer 5 full questions, selecting one full question from each module

Suggested Learning Resources:

Textbooks

1. B. S. Grewal: "Higher Engineering Mathematics", Khanna publishers, 44th Ed.2018
2. E. Kreyszig: "Advanced Engineering Mathematics", John Wiley & Sons, 10th Ed. (Reprint), 2016.

Reference Books:

1. V. Ramana: "Higher Engineering Mathematics" McGraw-Hill Education, 11th Ed.
2. Srimanta Pal & Subodh C. Bhunia: "Engineering Mathematics" Oxford University Press, 3rd Reprint, 2016.
3. N.P Bali and Manish Goyal: "A textbook of Engineering Mathematics" Laxmi Publications, Latest edition.
4. C. Ray Wylie, Louis C. Barrett: "Advanced Engineering Mathematics" McGraw – Hill Book Co.Newyork, Latest ed.
5. Gupta C.B, Sing S.R and Mukesh Kumar: "Engineering Mathematic for Semester I and II", Mc-Graw Hill Education(India) Pvt. Ltd 2015.
6. H.K.Dass and Er. Rajnish Verma: "Higher Engineering Mathematics" S.Chand Publication (2014).
7. James Stewart: "Calculus" Cengage publications, 7th edition, 4th Reprint 2019

Weblinks and Video Lectures (e-Resources):

1. http://www.class-central.com/subject/math(MOOCs)
2. http://academicearth.org/
3. http://www.bookstreet.in.
4. VTU e-Shikshana Program
5. VTU EDUSAT Program

Activity Based Learning (Suggested Activities in Class)/ Practical Based learning

- Quizzes
- Assignments
- Seminars

DATA STRUCTURES AND APPLICATIONS			
Course Code:	**21CS32**	CIE Marks	50
Teaching Hours/Week (L:T:P: S)	3:0:2:0	SEE Marks	50
Total Hours of Pedagogy	40 T + 20 P	Total Marks	100
Credits	04	Exam Hours	03

Course Objectives:

CLO 1. Explain the fundamentals of data structures and their applications essential for implementing solutions to problems.

CLO 2. Illustrate representation of data structures: Stack, Queues, Linked Lists, Trees and Graphs.

CLO 3. Design and Develop Solutions to problems using Arrays, Structures, Stack, Queues, Linked Lists.
CLO 4. Explore usage of Trees and Graph for application development.
CLO 5. Apply the Hashing techniques in mapping key value pairs.

Teaching-Learning Process (General Instructions)

These are sample Strategies, which teachers can use to accelerate the attainment of the various course outcomes.

1. Lecturer method (L) need not to be only traditional lecture method, but alternative effective teaching methods could be adopted to attain the outcomes.
2. Use of Video/Animation to explain functioning of various concepts.
3. Encourage collaborative (Group Learning) Learning in the class.
4. Ask at least three HOT (Higher order Thinking) questions in the class, which promotes critical thinking.
5. Adopt Problem Based Learning (PBL), which fosters students' Analytical skills, develop design thinking skills such as the ability to design, evaluate, generalize, and analyze information rather than simply recall it.
6. Introduce Topics in manifold representations.
7. Show the different ways to solve the same problem and encourage the students to come up with their own creative ways to solve them.
8. Discuss how every concept can be applied to the real world - and when that's possible, it helps improve the students' understanding.

Module-1

Introduction: Data Structures, Classifications (Primitive & Non-Primitive), Data structure operations (Traversing, inserting, deleting, searching, and sorting). Review of Arrays. Structures: Array of structures Self-Referential Structures.
Dynamic Memory Allocation Functions. Representation of Linear Arrays in Memory, dynamically allocated arrays and Multidimensional Arrays.
Demonstration of representation of Polynomials and Sparse Matrices with arrays.

Textbook 1: Chapter 1: 1.2, Chapter 2: 2.2 - 2.7, Text Textbook 2: Chapter 1: 1.1 - 1.4, Chapter 3: 3.1 - 3.3, 3.5, 3.7, Chapter 4: 4.1 - 4.9, 4.14 Textbook 3: Chapter 1: 1.3

Laboratory Component:

1. Design, Develop and Implement a menu driven Program in C for the following Array Operations
   a. Creating an Array of N Integer Elements
   b. Display of Array Elements with Suitable Headings
   c. Exit.

   Support the program with functions for each of the above operations.

2. Design, Develop and Implement a menu driven Program in C for the following Array operations
   a. Inserting an Element (ELEM) at a given valid Position (POS)
   b. Deleting an Element at a given valid Position POS)
   c. Display of Array Elements
   d. Exit.

   Support the program with functions for each of the above operations.

Teaching-Learning Process	Problem based learning (Implementation of different programs to illustrate application of arrays and structures. https://www.youtube.com/watch?v=3Xo6P_V-qns&t=201s https://ds2-iiith.vlabs.ac.in/exp/selection-sort/index.html https://ds1-iiith.vlabs.ac.in/data-structures-1/List%20of%20experiments.html

Module-2	
Stacks: Definition, Stack Operations, Array Representation of Stacks, Stacks using Dynamic Arrays. Different representation of expression. Stack Applications: Infix to postfix conversion, Infix to prefix conversion, evaluation of postfix expression, recursion. **Queues:** Definition, Array Representation of Queues, Queue Operations, Circular Queues, Queues and Circular queues using Dynamic arrays, Dequeues, Priority Queues. **Textbook 1: Chapter 3: 3.1 -3.4, 3.6 Textbook 2: Chapter 6: 6.1 -6.4, 6.5, 6.7-6.13**	
Laboratory Component: 1. Design, Develop and Implement a menu driven Program in C for the following operations on STACK of Integers (Array Implementation of Stack with maximum size MAX) a. *Push* an Element on to Stack b. *Pop* an Element from Stack c. Demonstrate *Overflow* and *Underflow* situations on Stack d. Display the status of Stack e. Exit Support the program with appropriate functions for each of the above operations 2. Design, Develop and Implement a Program in C for the following Stack Applications a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^ b. Solving Tower of Hanoi problem with n disks	
Teaching-Learning Process	Active Learning, Problem based learning https://nptel.ac.in/courses/106/102/106102064/ https://ds1-iiith.vlabs.ac.in/exp/stacks-queues/index.html
Module-3	
Linked Lists: Definition, classification of linked lists. Representation of different types of linked lists in Memory, Traversing, Insertion, Deletion, Searching, Sorting, and Concatenation Operations on Singly linked list, Doubly Linked lists, Circular linked lists, and header linked lists. Linked Stacks and Queues. Applications of Linked lists – Polynomials, Sparse matrix representation. Programming Examples. **Textbook 1: Chapter 4: 4.1 – 4.4, 4.5.2, 4.7, 4.8, Textbook 2: Chapter 5: 5.1 – 5.9**	
Laboratory Component: 1. Singly Linked List (SLL) of Integer Data a. Create a SLL stack of N integer. b. Display of SLL c. Linear search. Create a SLL queue of N Students Data Concatenation of two SLL of integers. 2. Design, Develop and Implement a menu driven Program in C for the following operationson Doubly Linked List (DLL) of Professor Data with the fields: ID, Name, Branch, Area of specialization a. Create a DLL stack of N Professor's Data. b. Create a DLL queue of N Professor's Data Display the status of DLL and count the number of nodes in it.	
Teaching-Learning Process	MOOC, Active Learning, Problem solving based on linked lists. https://nptel.ac.in/courses/106/102/106102064/ https://ds1-iiith.vlabs.ac.in/exp/linked-list/basics/overview.html https://ds1-iiith.vlabs.ac.in/List%20of%20experiments.html https://ds1-iiith.vlabs.ac.in/exp/linked-list/basics/overview.html https://ds1-iiith.vlabs.ac.in/List%20of%20experiments.html

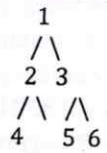
Module-4

Trees 1: Terminologies, Binary Trees, Properties of Binary trees, Array and linked Representation of Binary Trees, Binary Tree Traversals - Inorder, postorder, preorder; Threaded binary trees, Binary Search Trees – Definition, Insertion, Deletion, Traversal, and Searching operation on Binary search tree. Application of Trees-Evaluation of Expression.

Textbook 1: Chapter 5: 5.1 –5.5, 5.7; Textbook 2: Chapter 7: 7.1 – 7.9

Laboratory Component:

1. Given an array of elements, construct a complete binary tree from this array in level order fashion. That is, elements from left in the array will be filled in the tree level wise starting from level 0. Ex: Input :
   arr[] = {1, 2, 3, 4, 5, 6}
   Output : Root of the following tree



2. Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers
   a. Create a BST of N Integers
   b. Traverse the BST in Inorder, Preorder and Post Order

Teaching-Learning Process	Problem based learning http://www.nptelvideos.in/2012/11/data-structures-and-algorithms.html https://ds1-iiith.vlabs.ac.in/exp/tree-traversal/index.html https://ds1-iiith.vlabs.ac.in/exp/tree-traversal/depth-first-traversal/dft-practice.html

Module-5

Trees 2: AVL tree, Red-black tree, Splay tree, B-tree.

Graphs: Definitions, Terminologies, Matrix and Adjacency List Representation of Graphs, Traversal methods: Breadth First Search and Depth FirstSearch.

Hashing: Hash Table organizations, Hashing Functions, Static and Dynamic Hashing.

Textbook 1: Chapter 10:10.2, 10.3, 10.4, Textbook 2:7.10 – 7.12, 7.15 Chapter 11: 11.2, Textbook 1: Chapter 6 : 6.1–6.2, Chapter 8 : 8.1-8.3, Textbook 2: 8.1 – 8.3, 8.5, 8.7

Textbook 3: Chapter 15:15.1, 15.2,15.3, 15.4,15.5 and 15.7

Laboratory Component:

1. Design, Develop and implement a program in C for the following operations on Graph (G) of cities
   a. Create a Graph of N cities using Adjacency Matrix.
   b. Print all the nodes reachable from a given starting node in a diagraph using DFS/BFS method.

2. Design and develop a program in C that uses Hash Function H:K->L as H(K)=K mod m(reminder method) and implement hashing technique to map a given key K to the address space L. Resolve the collision (if any) using linear probing.

Teaching-Learning Process	NPTL, MOOC etc. courses on trees and graphs.

	http://www.nptelvideos.in/2012/11/data-structures-and-algorithms.html

Course Outcomes (Course Skill Set)

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

CO 1. Identify different data structures and their applications.
CO 2. Apply stack and queues in solving problems.
CO 3. Demonstrate applications of linked list.
CO 4. Explore the applications of trees and graphs to model and solve the real-world problem.
CO 5. Make use of Hashing techniques and resolve collisions during mapping of key value pairs

Assessment Details (both CIE and SEE)

The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. The minimum passing mark for the CIE is 40% of the maximum marks (20 marks). A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each subject/ course if the student secures not less than 35% (18 Marks out of 50) in the semester-end examination (SEE), and a minimum of 40% (40 marks out of 100) in the sum total of the CIE (Continuous Internal Evaluation) and SEE (Semester End Examination) taken together

Continuous Internal Evaluation:

Three Unit Tests each of **20 Marks (duration 01 hour)**

1. First test at the end of 5th week of the semester
2. Second test at the end of the 10th week of the semester
3. Third test at the end of the 15th week of the semester

Two assignments each of **10 Marks**

4. First assignment at the end of 4th week of the semester
5. Second assignment at the end of 9th week of the semester

Practical Sessions need to be assessed by appropriate rubrics and viva-voce method. This will contribute to **20 marks**.

- Rubrics for each Experiment taken average for all Lab components – 15 Marks.
- Viva-Voce– 5 Marks (more emphasized on demonstration topics)

The sum of three tests, two assignments, and practical sessions will be out of 100 marks and will be **scaled down to 50 marks**

(to have a less stressed CIE, the portion of the syllabus should not be common /repeated for any of the methods of the CIE. Each method of CIE should have a different syllabus portion of the course).

CIE methods /question paper has to be designed to attain the different levels of Bloom's taxonomy as per the outcome defined for the course.

Semester End Examination:

Theory SEE will be conducted by University as per the scheduled timetable, with common question papers for the subject (**duration 03 hours**)

1. The question paper will have ten questions. Each question is set for 20 marks. Marks scored shall be proportionally reduced to 50 marks
2. There will be 2 questions from each module. Each of the two questions under a module (with a maximum of 3 sub-questions), **should have a mix of topics** under that module.

The students have to answer 5 full questions, selecting one full question from each module

Suggested Learning Resources:

Textbooks:

1. Ellis Horowitz and Sartaj Sahni, Fundamentals of Data Structures in C, 2nd Ed, Universities Press, 2014.

2. Seymour Lipschutz, Data Structures Schaum's Outlines, Revised 1st Ed, McGraw Hill, 2014.
3. Reema Thareja, Data Structures using C, 3rd Ed, Oxford press, 2012.

Reference Books:

1. Gilberg and Forouzan, Data Structures: A Pseudo-code approach with C, 2nd Ed, Cengage Learning,2014.
2. Jean-Paul Tremblay & Paul G. Sorenson, An Introduction to Data Structures with Applications,2nd Ed, McGraw Hill, 2013
3. A M Tenenbaum, Data Structures using C, PHI, 1989
4. Robert Kruse, Data Structures and Program Design in C, 2nd Ed, PHI, 1996.

Weblinks and Video Lectures (e-Resources):

1. http://elearning.vtu.ac.in/econtent/courses/video/CSE/06CS35.html
2. https://nptel.ac.in/courses/106/105/106105171/
3. http://www.nptelvideos.in/2012/11/data-structures-and-algorithms.html

Activity Based Learning (Suggested Activities in Class)/ Practical Based learning

- Real world problem solving using group discussion.
- Back/Forward stacks on browsers.
- Undo/Redo stacks in Excel or Word.
- Linked list representation of real-world queues -Music player, image viewer

ALVA'S Education Foundation

Alva's Institute of Engineering & Technology

Shobhavana Campus, Mijar, Moodbidri, D.K – 574225

Phone: 08258-262725, Fax: 08258-262726

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

SEMESTER III

Course Code:**18CS32** | Course Name: **Data Structures and Applications**

Course Teacher: **Prof. Soundarya B C**

Course Outcomes: After studying this course, students will be able to:

CO Numbers	Course Outcomes	Blooms Level	Target Level
18CS32.1	**Observe** data structure definition, classification and data structure operations and **Demonstrate** dynamic memory allocations.	Apply (L3)	2
18CS32.2	**Develop** the operations of stack queues and **Implement** the algorithms for stack applications.	Apply (L3) Analyze (L4)	2
18CS32.3	**Explain** the SLL, DLL and CLL by its operations and **Implement** the algorithms for its applications such as polynomials and sparse matrix.	Apply (L3)	2
18CS32.4	**Construct** the operations of trees and **Implement** the algorithms for the given problems using binary trees, BST.	Apply (L3)	2
18CS32.5	**Manipulate** the applications of graphs, methods for hash table organizations.	Analyze (L4)	2

CO-PO/CO-PSO Mapping Matrix:

CO Numbers	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO 1	PSO2	PSO3
18CS32.1	2	1	2		1					1			2	1	
18CS32.2	2	2	2		1					2			2	1	
18CS32.3	2	2	2	2	1					2			2	2	
18CS32.4	2	1	2	1	1					2			2	2	
18CS32.5	2	1	1		1					2			2	2	
AVG	2	1.4	1.8	1.5	1					1.8			2	1.6	

ALVA'S
Education Foundation®

Alva's Institute of Engineering & Technology

Shobhavana Campus, Mijar, Moodbidri, D.K – 574225

Phone: 08258-262725, Fax: 08258-262726

CO-PO/CO-PSO Mapping Matrix Justification: I want my students to			
CO	**POs**	**Level**	**Justification**
18CS32.1	PO1	2	Moderate knowledge of basic data structures.
	PO2	1	Basic analysis of array applications
	PO3	2	Implementation of simple array applications.
	PO5	1	Usage of virtual lab for arrays
	PO10	1	Conduction of hands on training and assessment
	PSO1	2	Moderate Knowledge of arrays and structures
	PSO2	1	Basic mathematical problems are solved using arrays
18CS32.2	PO1	2	Moderate knowledge of data structures
	PO2	2	Moderate analysis of stack and queues
	PO3	2	Implementation of applications of stacks
	PO5	1	Usage of virtual lab for stacks and queues
	PO10	1	Conduction of hands on training and assessment
	PSO1	2	Moderate knowledge of structures, stacks and types of queues
	PSO2	1	Basic mathematical problems are solved using stack and queue
18CS32.3	PO1	2	Moderate knowledge of pointers, stacks, queues and dynamic memory allocations.
	PO2	2	Moderate analysis of SLL, DLL and CLL operations
	PO3	2	Implementation of application such as polynomials and sparse.
	PO4	2	Analyzing working of polynomial additions and sparse matrix multiplication
	PO5	1	Usage of virtual lab for SLL, DLL and CLL.
	PO10	1	Conduction of hands on training and assessment
	PSO1	2	Moderate knowledge of pointers, stacks, queues and dynamic memory allocation
	PSO2	2	Moderate analysis of SLL, DLL and CLL operations
18CS32.4	PO1	2	Moderate knowledge of pointers , linked lists and arrays
	PO2	1	Basic analysis of types of binary trees and its operations
	PO3	2	Implementation of tree traversals and binary search trees
	PO4	1	Analyzing working of binary tree traversals and application of trees
	PO5	1	Usage of virtual lab for implementing the tree traversals and evaluation of expression
	PO10	1	Conduction of hands training and assessment
	PSO1	2	Moderate knowledge of pointers, linked lists and arrays
	PSO2	2	Basic analysis of types of binary trees and its operations
18CS32.5	PO1	1	Moderate knowledge of pointers, linked lists and arrays
	PO2	1	Basic analysis of graph traversals such as BFS, DFS and file handling
	PO3	1	Implementation of BFS and DFS, sorting, searching, and hashing
	PO5	1	Virtual lad usage for implementing the graphs traversal
	PO10	1	Conduction of hands on training and assessment
	PSO1	2	Moderate knowledge of pointers, linked lists and arrays

Course Teacher
ignature with date

IQAC Member
Signature with date

IQAC Chairman
Signature with date

ALVA'S INSTITUTE OF ENGINEERING & TECHNOLOGY

(Unit of Alva's Education Foundation (R), Moodbidri)
Affiliated to Visvesvaraya Technological University, Belagavi & Approved by AICTE, New Delhi. Recognized by Government of Karnataka.
A+, Accredited by NACC & NBA (ECE & CSE)
Shobhavana Campus, MIJAR-574225, Moodbidri, D.K., Karnataka
Ph: 08258-262725; Mob:722262724,7026262725,mail:principalaiet08@gmail.com

Department of Computer Science and Design

Date 31-10-2022

Student list for the academic year 2022-23

SL. NO	USN	Student Name
1	4AL21CG001	ABDUL AZEEZ
2	4AL21CG002	ABHISHEK E B
3	4AL21CG003	ABHISHEK N C
4	4AL21CG004	ADARSH BHAVIMANE
5	4AL21CG005	ALEX TAJENJAM
6	4AL21CG006	ANUSHA N T
7	4AL21CG007	ASHISH BHUTKURI
8	4AL21CG008	ASHWIN K S
9	4AL21CG009	B PRAKASH
10	4AL21CG010	BALAJI SATISH KULKARNI
11	4AL21CG011	BHARAT
12	4AL21CG012	BHAVISH M K
13	4AL21CG013	CHANDAN KUMAR M
14	4AL21CG014	CHINMAY T N
15	4AL21CG015	DARSHAN RAI
16	4AL21CG016	DEEKSHITH ACHARYA
17	4AL21CG017	DEEPASHREE G NAIK
18	4AL21CG018	DEVADIGA ROSHNI NARAYAN
19	4AL21CG019	DHANUSRI R
20	4AL21CG020	DHANUSH A S
21	4AL21CG021	DHANUSH SHENOY
22	4AL21CG022	DHEERAJ SHETTY
23	4AL21CG023	ESHWARI K C
24	4AL21CG024	GHRUTHAVARSHA K G
25	4AL21CG025	GIREESH H
26	4AL21CG026	H M TUSHAR
27	4AL21CG027	HARSHITH
28	4AL21CG028	HEMANTH MUNAVALLI
29	4AL21CG029	JAHNAVI V
30	4AL21CG030	JAYAPRAKASH P
31	4AL21CG031	KARTHIK
32	4AL21CG032	KARTHIK P NAIK
33	4AL21CG033	KIRAN KUMAR K
34	4AL21CG034	LAKSHMISHA K A
35	4AL21CG035	LIKHITH L
36	4AL21CG036	MANOJ M
37	4AL21CG037	MOHAMMED SAAD
38	4AL21CG038	NAKUL N
39	4AL21CG039	NARESH MAIBAM
40	4AL21CG040	NAYANA S M
41	4AL21CG041	NISHAAN H. SHETTY
42	4AL21CG042	PRATHVIRAJ K
43	4AL21CG043	RACHANA
44	4AL21CG044	RAKSHA SIDDESH G
45	4AL21CG045	RAKSHITHA
46	4AL21CG046	RANJEETH P JAIN
47	4AL21CG047	RIDHI R HEGDE
48	4AL21CG048	SANDHYA MOOLYA
49	4AL21CG049	SHARVARI M S
50	4AL21CG051	SHIBANI
51	4AL21CG052	SHIVANI
52	4AL21CG053	SHRAVYA
53	4AL21CG054	SHREEYA L
54	4AL21CG055	SINDHU N
55	4AL21CG056	SHREE GANESH M S
56	4AL21CG057	SUJAY KUMAR B ADOOR
57	4AL21CG058	SURAJ
58	4AL21CG059	SURAKSHA
59	4AL21CG060	VIKRAM
60	4AL21CG061	VIMALKUMAR U REVANKAR
61	4AL21CG062	VINITH KALIKAR
62	4AL21CG063	YASHUYADAV A

HOD

ALVA'S INSTITUTE OF ENGINEEIRNG & TECHNOLOGY,

MIJAR, MOODBIDRI – 574225

Department of Computer Science and Design

Academic Year 2022-23 (Odd Semester) First Internal Assessment Test

Subject: Data Structures and Applications (21CS32) **Semester: III**

Max. Marks: 20 **Date: 7/12/2022**

Faculty: Ms. Soundarya B C **Time: 3:00 PM to 4:00 PM**

Note: Answer any TWO full questions by selecting one full question from each part.

Question#		Questions	Marks	CO	BT/CL
		PART-A			
1	a.	What is Data Structure? Explain with a neat schemantic diagram classification of Data Structures.	5	CO3	L3
	b.	What is Dynamic Memory Allocation? Illustrate Dynamic Memory Allocation functions with syntax.	5	CO1	L3
		OR			
2	a.	Explain Self Referential Structure with syntax and Example.	5	CO3	L3
	b.	What is an array? Examine Insertion and Deletion operations with example.	5	CO1	L3
		PART-B			
3	a.	Construct the push and pop operations of stack using arrays.	6	CO2	L4
	b.	Assume A=1, B=2, C=3. Evaluate the following postfix expressions: a. A B + C – B A + C - + b. A B C + * C B A – + *	4	CO5	L4
		OR			
4	a.	Implement the operations of queue using arrays.	6	CO2	L4
	b.	Convert Infix to Prefix expression (A+B^C)*D+E^5	4	CO5	L4

CO1 **Observe** data structure definition, classification and data structure operations and **Demonstrate** dynamic memory allocations.

CO2 **Develop** the operations of stack queues and **Implement** the algorithms for stack applications.

CO3 **Explain** the SLL, DLL and CLL by its operations and **Implement** the algorithms for its applications such as polynomials and sparse matrix.

CO5 **Manipulate** the applications of graphs, methods for hash table organizations.

Signature of the faculty **IQAC Member** **IQAC Chairman**

12

ALVA'S INSTITUTE OF ENGINEERING&TECHNOLOGY, MOODBIDRI

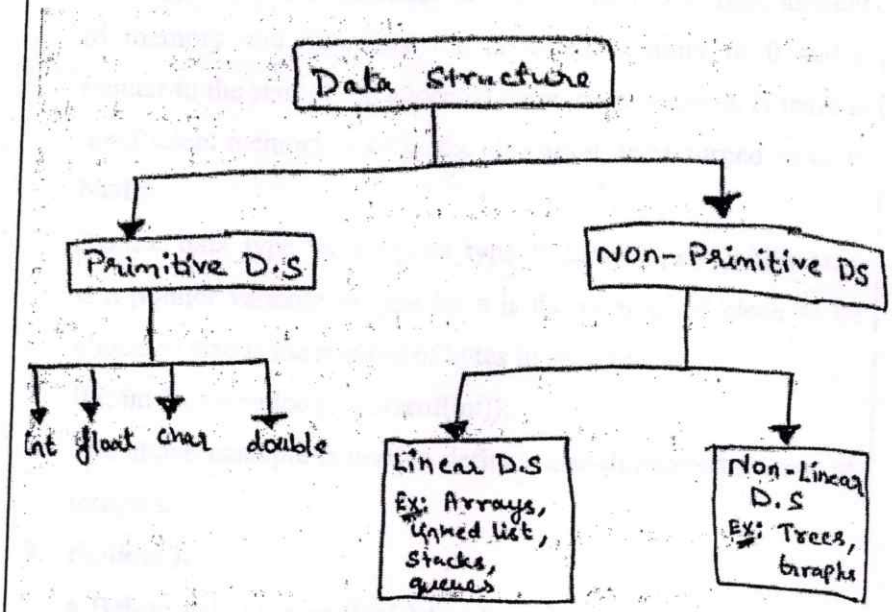
(A unit of Alva's Education Foundation)

III Semester B.E. I. A Test Examinations-I

7th December–2022

21CS32 – DATA STRUCTURES AND APPLICATIONS

SCHEME OF VALUATION

Questions		Details	Marks
		PART - A	
1	a	Definition - 1 Mark Data may be organized in many different ways. The logical or mathematical model of a particular organization of data is called a data structure. Data structures are generally classified into • Primitive data Structures • Non-primitive data Structures  1. **Primitive data Structures:** Primitive data structures are the fundamental data types which are supported by a programming language. Basic data types such as integer, real, character and Boolean are known as Primitive data Structures. These data types consists of characters that cannot be divided and hence they also called simple data types.	5

		2. Non- Primitive data Structures: Non-primitive data structures are those data structures which are created using primitive data structures. Examples of non-primitive data structures is the processing of complex numbers, linked lists, stacks, trees, and graphs. Based on the structure and arrangement of data, non-primitive data structures is further classified into 1. Linear Data Structure 2. Non-linear Data Structure	
	b	1. **malloc( ):** The function malloc allocates a user- specified amount of memory and a pointer to the start of the allocated memory is returned. If there is insufficient memory to make the allocation, the returned value is NULL. Syntax: data_type *x; x= (data_type *) malloc(size); Where, x is a pointer variable of data_type size is the number of bytes Ex: int *ptr; ptr = (int *) malloc(100*sizeof(int)); 2. **calloc( ):** The function calloc allocates a user- specified amount of memory and initializes the allocated memory to 0 and a pointer to the start of the allocated memory is returned. If there is insufficient memory to make the allocation, the returned value is NULL. Syntax: data_type *x; x= (data_type *) calloc(n, size); Where, x is a pointer variable of type int n is the number of block to be allocated size is the number of bytes in each block Ex: int *x x= calloc (10, sizeof(int)); The above example is used to define a one-dimensional array of integers. 3. **realloc( ):** • Before using the realloc( ) function, the memory should have been allocated using malloc( ) or calloc( ) functions. • The function relloc( ) resizes memory previously allocated by either mallor or calloc, which means, the size of the memory changes by extending or deleting the allocated memory. • If the existing allocated memory need to extend, the pointer value will not change. Syntax- data_type *x;	5

x= (data_type *) realloc(p, s);

4. **Free():** Frees the memory allocated by malloc(), calloc() and realloc().

   Syntax : Free();

OR

2 a

SELF-REFERENTIAL STRUCTURES

A self-referential structure is one in which one or more of its components is a pointer to itself. Self-referential structures usually require dynamic storage management routines (malloc and free) to explicitly obtain and release memory.

Consider as an example:

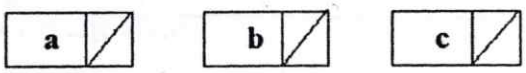
```
typedef struct {
        char data;
        struct list *link;
    } list;
```

Each instance of the structure **list** will have two components **data** and **link**.

- **Data:** is a single character,
- **Link:** link is a pointer to a list structure. The value of link is either the address in memory of an instance of list or the null pointer.

Consider these statements, which create three structures and assign values to their respective fields:

```
list item1, item2, item3;
item1.data = 'a';
item2.data = 'b';
item3.data = 'c';
item1.link = item2.1ink = item3.link = NULL;
```



5

b

Array Defintion – 1 Mark

Array is a homogeneous collection of elements.

5

4. Push( )

Function **push** checks whether stack is full. If it is, it calls stackFull( ), which prints an error message and terminates execution. When the stack is not full, increment top and assign item to stack [top].

```
void push(element item)
    { /* add an item to the global stack */
        if (top >= MAX_STACK_SIZE-1)
            stackFull();
        stack[++top] = item;
    }
```

5. Pop( )

Deleting an element from the stack is called pop operation. The element is deleted only from the top of the stack and only one element is deleted at a time.

```
element pop ( )
    { /*delete and return the top element from the stack */
        if (top == -1)
            return stackEmpty(); /*returns an error key */
        return stack[top--];
    }
```

6. stackFull( )

The **stackFull** which prints an error message and terminates execution.

```
void stackFull()
    {
        fprintf(stderr, "Stack is full, cannot add element");
        exit(EXIT_FAILURE);
    }
```

b

a. A B + C – B A + C –

Answer- 0

b. A B C + * C B A – + *

Answer - 20

4

OR

4 a

6

QUEUE OPERATIONS

Implementation of the queue operations as follows.

1. Queue Create

```
Queue CreateQ(maxQueueSize) ::=
        #define MAX_QUEUE_SIZE 100          /* maximum queue size */
        typedef struct
                {
                        int key;            /* other fields */
                } element;
        element queue[MAX_QUEUE_SIZE];
        int rear = -1;
        int front = -1;
```

2. Boolean IsEmptyQ(queue) ::= front ==rear

3. Boolean IsFullQ(queue) ::= rear == MAX_QUEUE_SIZE-1

In the queue, two variables are used which are **front** and **rear.** The queue increments **rear** in addq( ) and **front** in delete( ). The function calls would be
addq (item); and **item =delete( );**

4. addq(item)

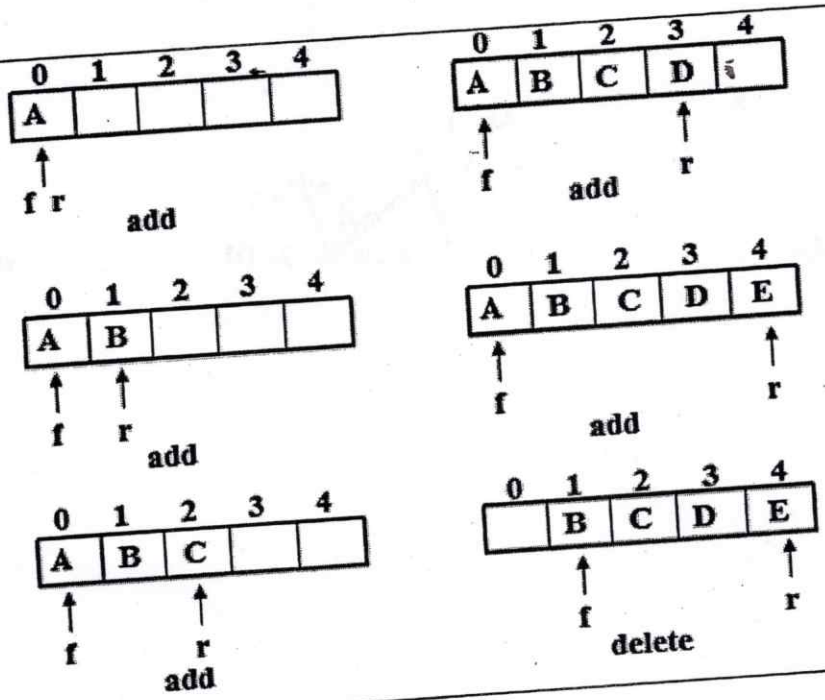
```
void addq(element item)
        {                       /* add an item to the queue */
                if (rear == MAX_QUEUE_SIZE-1)
                        queueFull();
                queue [++rear] = item;
        }
```

Program: Add to a queue

5. deleteq( )

```
element deleteq()
        {                       /* remove element at the front of the queue */
                if (front == rear)
                        return queueEmpty( );           /* return an error key */
                return queue[++front];
        }
```

Program: Delete from a queue



b Expression Conversion – 4 Mark

Expression = (A+B^C)*D+E^5

Step 1. Reverse the infix expression.

5^E+D*)C^B+A(

Step 2. Make Every '(' as ')' and every ')' as '('

5^E+D*(C^B+A)

Step 3. Convert expression to postfix form.

A+(B*C-(D/E-F)*G)*H

Expression	Stack	Output	Comment
5^E+D*(C^B+A)	Empty	-	Initial
^E+D*(C^B+A)	Empty	5	Print
E+D*(C^B+A)	^	5	Push
+D*(C^B+A)	^	5E	Push
D*(C^B+A)	+	5E^	Pop And Push
*(C^B+A)	+	5E^D	Print
(C^B+A)	+*	5E^D	Push
C^B+A)	+*(	5E^D	Push
^B+A)	+*(	5E^DC	Print
B+A)	+*(^	5E^DC	Push
+A)	+*(^	5E^DCB	Print
A)	+*(+	5E^DCB^	Pop And Push
)	+*(+	5E^DCB^A	Print
End	+*	5E^DCB^A+	Pop Until '('
End	Empty	5E^DCB^A+*+	Pop Every element

4

Signature of the faculty

IQAC Member

31/12/2023

IQAC Chairman

ALVA'S INSTITUTE OF ENGINEEIRNG & TECHNOLOGY,

MIJAR, MOODBIDRI – 574225

Department of Computer Science and Design

Academic Year 2022-23 (Odd Semester) Second Internal Assessment Test

Subject: Data Structures and Applications (21CS32) **Semester: III**

Max. Marks: 20 **Date: 16/01/2023**

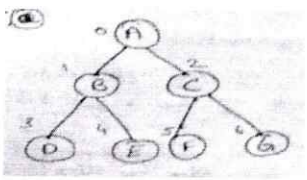
Faculty: Ms. Soundarya B C **Time: 3:00 PM to 4:00 PM**

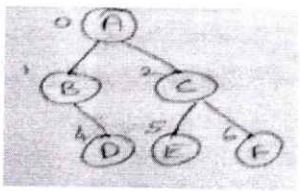
Note: Answer any TWO full questions by selecting one full question from each part.

Q#			Marks	CO	BT/CL
		PART-A			
1	a.	Write and Explain how do you implement the operations of stack using Singly Linked List with the help of C Program.	10	CO3	L3
		OR			
2	a.	With a C function Demonstrate how to insert a node at front and delete node at end in a Doubly Linked List. Implement an algorithm to add two polynomials and For the given sparse matrix, give the linked list representation.	10	CO1	L3

$$\begin{vmatrix} 0 & 1 & 2 \\ 3 & 0 & 4 \\ 0 & 0 & 0 \end{vmatrix}$$

		PART-B			
3	a.	What is tree? Define any 5 terminologies of tree with example.	6	CO1	L4
	b.	Illustrate the representation of binary tree.	4	CO4	L3
		OR			
4	a.	Represent the given tree in using 1. Linked list representation 2. Left child Right Sibling	6	CO1	L4





	b.	Define Binary Tree and Illustrate the following 1. Complete Binary Tree 2. Skewed Binary Tree	4	CO4	L3

C01 Observe data structure definition, classification and data structure operations and **Demonstrate** dynamic memory allocations.

C03 Explain the SLL, DLL and CLL by its operations and **Implement** the algorithms for its applications such as polynomials and sparse matrix.

C04 Construct the operations of trees and **Implement** the algorithms for the given problems using binary trees, BST.

Signature of the faculty **IQAC Member** **IQAC Chairman**

ALVA'S INSTITUTE OF ENGINEERING&TECHNOLOGY, MOODBIDRI

(A unit of Alva's Education Foundation)

III Semester B.E. ( ) IA Test Examinations-III

16th March–2023

21CS32 – DATA STRUCTURES AND APPLICATIONS

SCHEME OF VALUATION

Questions		Details	Marks
		PART - A	

1	a	Post-order
		void postOrder(node root)
		{ if(root==NULL) return;
		postOrder(root->llink);
		postOrder(root->rlink);
		Pf("%d", root->info);
		}
		Inorder
		void inOrder(node root)
		{ if(root==NULL) return;
		inOrder(root->llink);
		Pf("%d", root->info);
		inOrder(root->rlink);
		}
		Preorder
		void preOrder(node root)
		{ if(root==NULL) return;
		Pf("%d", root->info);
		preOrder(root->llink);
		preOrder(root->rlink);
		}	

	b	Defn – 1M
		Node search(int item, Node root)
		{ if(root==NULL) return root;
		if(item==root->info) return root;
		if(item<root->info)
		else return search(item, root->llink);
		} return search(item, root->rlink)
			7
			3

OR

2 a

Inorder - 4, 2, 5, 1, 3

preorder - 1, 2, 4, 5, 3

postorder - 4, 5, 2, 3, 1

b) Inorder -
B D A G F C H E I

A B D C E G F H I.

D B G E H I F C A

b

7

```
Node insert(int item, node root)
{
  node temp, cur, prev;
  temp = (node*)malloc(sizeof(node));
  temp->info = item;
  temp->llink = temp->rlink = NULL;
  if(root == NULL) return temp;
  prev = null;
  cur = root;
  while(cur != NULL)
  {
    prev = cur;
    if(item == cur->info)
    {
      Pf("Duplicates are not allowed");
      return root;
    }
  if(item < cur->info)
    cur = cur->llink;
  else
    cur = cur->rlink;
}
```

PART – B

3 a

Division method

The hash function depends upon the remainder of division. Typically the divisor is the table length.

eg: 54, 72, 89, 37. size is 10.

h(key) = record % SIZE

h(54) = 54 % 10 = 4

h(72) = 72 % 10 = 2

h(89) = 89 % 10 = 9.

h(37) = 37 % 10 = 7.

Mid-Square method.

Here 'K' is squared.

A number 'l' in the middle of K^2 is selected by removing the digits from both ends.

Folding method.

$$h(k) = K1 + K2 + K3 + \dots + Kn$$

K = 123987234876

$K_1 = 12$, $K_2 = 39$, $K_3 = 87$, $K_4 = 23$, $K_5 = 218$, $K_6 = 76$.

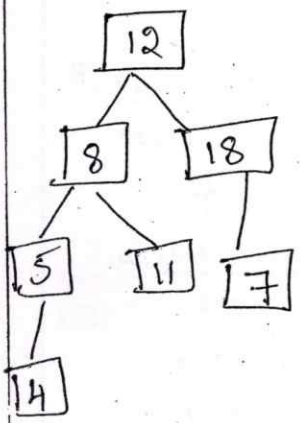
= 12 + 39 + 87 + 23 + 218 + 76

h(k) = 285

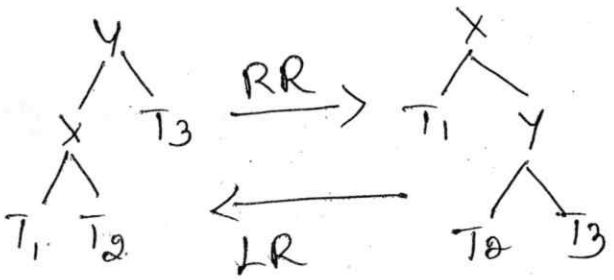
b

Insertion operation of AVL tree

AVL Tree is a self-balancing BST where the diff b/w heights of left & right subtrees cannot be more than one for all nodes.



1. Left rotation
2. Right rotation



eg:-

4

OR

4 a

Graph

Formally, a graph G is defined as a pair of two sets V & E, denoted by

$$G = (V, E)$$

1. Vertex
2. Edge.
3. Path.
4. Undirected edge.
5. Directed edge.

6

① Vertex : A vertex is a synonym for node. A vertex is represented by a circle

Eg: (1) (2) (3) — nodes 1, 2, 3 are vertices.

② Edge : An arc or a line joining two vertices say u and v is called an edge.

Eg: (1)——(2)

a) Undirected edge : No direction b/w 2 vertices

Eg: (1)——(2)

* Denoted by an ordered pair (1,2)
* (1,2) is same as (2,1) in an undirected graph

b) Directed edge : a direction exists b/w 2 vertices Eg: (1)——→(2)

b

Types of Hashing Techniques

a) Static Hashing b) Dynamic hashing

a) Static Hashing : is a technique in which the table (bucket) size remains the same (fixed during compilation time) is called static hashing.

* Various techniques of static hashing are linear probing, chaining.
* As the size is fixed, this type of hashing cannot handle overflow of elements (collision) efficiently

Eg: Elements to be stored: 24, 93, 45

h(24) = 24%10 = 4

h(93) = 93%10 = 3

h(45) = 45%10 = 5

b) Dynamic hashing

This is an hashing technique in which the bucket size is not fixed. It can grow or shrink according to the increase or decrease of records.

4

ALVA'S INSTITUTE OF ENGINEEIRNG & TECHNOLOGY,

MIJAR, MOODBIDRI – 574225

Department of Computer Science and Design

Academic Year 2022-23 (Odd Semester) Third Internal Assessment Test

Subject: Data Structures and Applications (21CS32) **Semester: III**

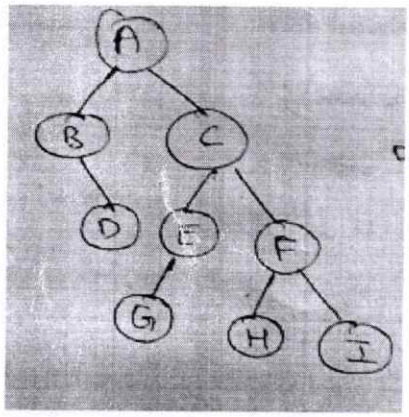
Max. Marks: 20 **Date: 16/03/2023**

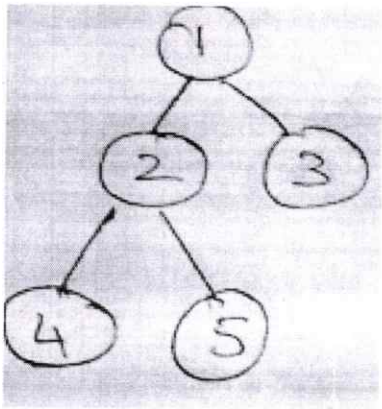
Faculty: Ms. Soundarya B C **Time: 3:00 PM to 4:00 PM**

Note: Answer any TWO full questions by selecting one full question from each part.

PART-A

		Marks	CO	BT/CL
1 a.	Define traversal. With a function Illustrate the traversal techniques.	7	CO4	L4
b.	What is BST? Implement a search function to search BST for an item.	3	CO1	L3
	OR			
a.	Define Degree of the tree. Write In order, Pre-order and Post order traversal for the below trees.	7	CO4	L4





b.	Write a function to insert an element in a BST.	3	CO1	L3

PART-B

3 a.	Explain different types of hash functions with example.	6	CO5	L3
b.	Illustrate Insertion operation of AVL Tree.	4	CO5	L4
	OR			
4 a.	Define Tree. List and explain any 5 graph terminologies.	6	CO5	L3
b.	Explain in detail different types of hashing techniques.	4	CO5	L4

1. **Observe** data structure definition, classification and data structure operations and **Demonstrate** dynamic memory allocations. (L3)
2. **Construct** the operations of trees and **Implement** the algorithms for the given problems using binary trees, BST. (L3)
3. **Manipulate** the applications of graphs, methods for hash table organizations. (L4)

Signature of the faculty **IQAC Member** **IQAC Chairman**

ALVA'S INSTITUTE OF ENGINEERING&TECHNOLOGY, MOODBIDRI

(A unit of Alva's Education Foundation)

III Semester B.E. I. A Test Examinations-II

16th December–2023

21CS32 – DATA STRUCTURES AND APPLICATIONS

SCHEME OF VALUATION

Questions		Details	Marks
		PART - A	
1	a	Program Defining structure 2.5 M Push() – 2.5 M Pop() – 2.5 M Display() – 2.5 M	10
		OR	

2	a	C Function for Insert with representation - 3 M
		Insertion into a doubly linked list
		Insertion into a doubly linked list is fairly easy. Assume there are two nodes, node and newnode, node may be either a header node or an interior node in a list. The function dinsert performs the insertion operation in constant time.
		`void dinsert(nodePointer node, nodePointer newnode)`
		`{/* insert newnode to the right of node */`
		`newnode→llink = node;`
		`newnode→rlink = node→rlink;`
		`node→rlink→llink = newnode;`
		`node→rlink = newnode;`
		`}`
		Program: Insertion into a doubly linked circular list
		C Function for Delete with representation – 3 M
			6

Deletion from a doubly linked list

Deletion from a doubly linked list is equally easy. The function *ddelete* deletes the node deleted from the list pointed to by node.

To accomplish this deletion, we only need to change the link fields of the nodes that precede (deleted→llink→rlink) and follow (deleted→rlink→llink) the node we want to delete.

```
void ddelete(nodePointer node, nodePointer deleted)
    {/* delete from the doubly linked list */
        if (node == deleted)
            printf("Deletion of header node not permitted.\n");
        else {
            deleted→llink→rlink = deleted→rlink;
            deleted→rlink→llink = deleted→llink;
            free(deleted);
          }
    }
```

Program: Deletion from a doubly linked circular list

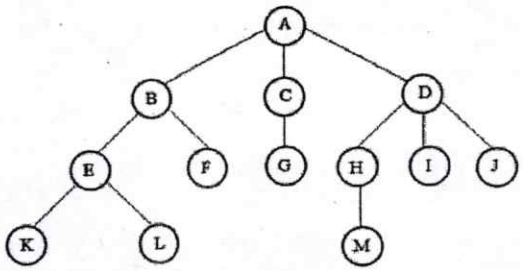
b

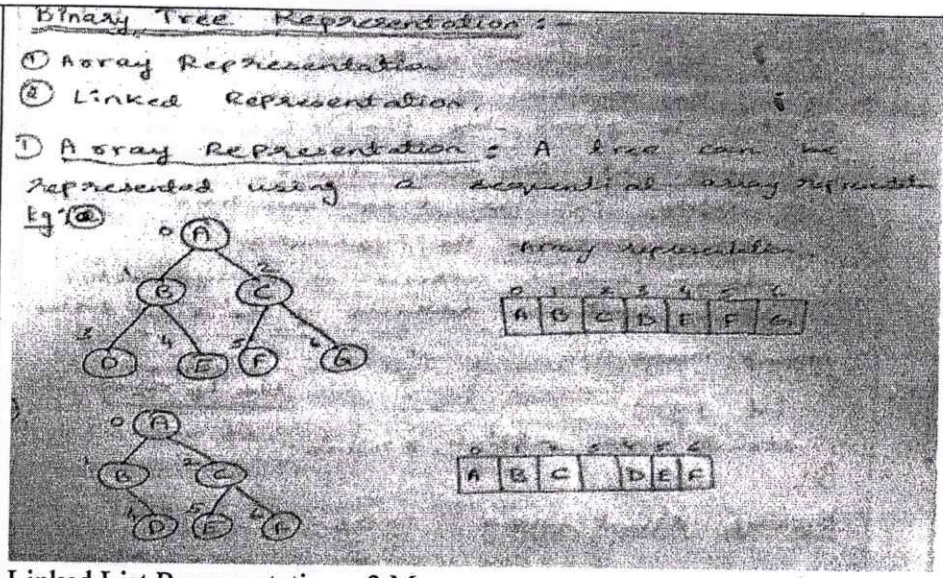
Algorithm – 2 Mark

```
polyPointer padd(polyPointer a, polyPointer b)
{/* return a polynomial which is the sum of a and b */
  polyPointer c, rear, temp;
  int sum;
  MALLOC(rear, sizeof(*rear));
  c = rear;
  while (a && b)
     switch (COMPARE(a→expon,b→expon)) {
        case -1: /* a→expon < b→expon */
              attach(b→coef,b→expon,&rear);
              b = b→link;
              break;
        case 0: /* a→expon = b→expon */
              sum = a→coef + b→coef;
              if (sum) attach(sum,a→expon,&rear);
              a = a→link;  b = b→link; break;
        case 1: /* a→expon > b→expon */
              attach(a→coef,a→expon,&rear);
              a = a→link;
     }
  /* copy rest of list a and then list b */
  for (; a; a = a→link) attach(a→coef,a→expon,&rear);
  for (; b; b = b→link) attach(b→coef,b→expon,&rear);
  rear→link = NULL;
  /* delete extra initial node */
  temp = c; c = c→link;  free(temp);
  return c;
}
```

Linked List representation of sparse matrix – 2 M

PART - B 4

3	a	Definition – 1 M Terminologies – 5 M **DEFINITION** A *tree* is a finite set of one or more nodes such that • There is a specially designated node called *root*. • The remaining nodes are partitioned into $n >= 0$ disjoint set $T_1,\ldots,T_n$, where each of these sets is a tree. $T_1,\ldots,T_n$ are called the *subtrees* of the root.  **TERMINOLOGY** • ***Node:*** The item of information plus the branches to other nodes • ***Degree:*** The number of subtrees of a node • ***Degree of a tree:*** The maximum of the degree of the nodes in the tree. • ***Terminal nodes (or leaf):*** nodes that have degree zero or node with no successor • ***Nonterminal nodes***: nodes that don't belong to terminal nodes. • ***Parent and Children:*** Suppose N is a node in T with left successor S1 and right successor S2, then N is called the Parent (or father) of S1 and S2. Here, S1 is called left child (or Son) and S2 is called right child (or Son) of N. • ***Siblings:*** Children of the same parent are said to be siblings. • ***Edge:*** A line drawn from node N of a T to a successor is called an edge • ***Path:*** A sequence of consecutive edges from node N to a node M is called a path. • ***Ancestors of a node:*** All the nodes along the path from the root to that node. • ***The level of a node:*** defined by letting the root be at level zero. If a node is at level l, then it children are at level $l+1$. • ***Height (or depth):*** The maximum level of any node in the tree	6
	b	Array Representation – 2 M	4



Linked List Representation – 2 M

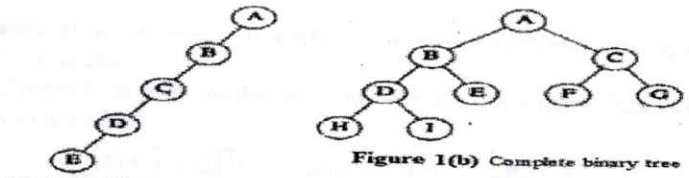
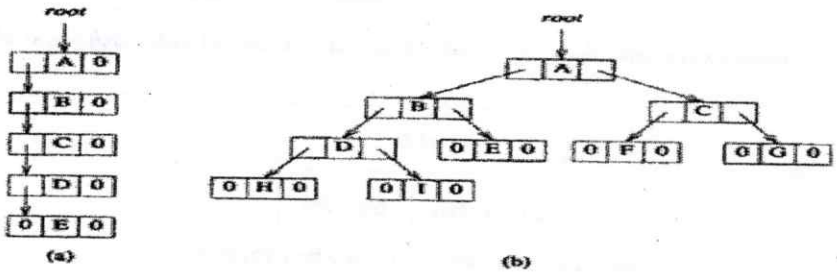


Figure 1(a) Skewed binary tree

Figure 1(b) Complete binary tree



Linked representation of the binary tree

		OR	
4	a	1. Linked list representation – 3 M 2. Left child Right Sibling - 3 M	6

List Representation:

The tree can be represented as a List. The tree of figure (A) could be written as the list.

(A (B (E (K, L), F), C (G), D (H (M), I, J)))

- The information in the root node comes first.
- The root node is followed by a list of the subtrees of that node.

Tree node is represented by a memory node that has fields for the data and pointers to the tree node's children

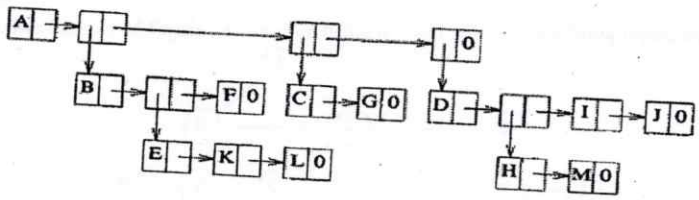
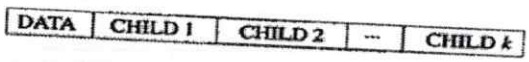
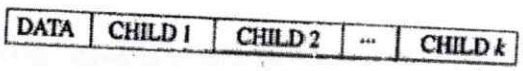


Figure (B): List representation of the tree of figure (A)

Since the degree of each tree node may be different, so memory nodes with a varying number of pointer fields are used.
For a tree of degree k, the node structure can be represented as below figure. Each child field is used to point to a subtree.



Since the degree of each tree node may be different, so memory nodes with a varying number of pointer fields are used.
For a tree of degree k, the node structure can be represented as below figure. Each child field is used to point to a subtree.



Left Child-Right Sibling Representation

The below figure show the node structure used in the left child-right sibling representation

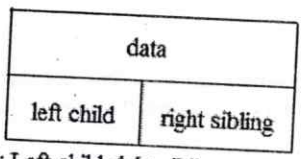


Figure (c): Left child right sibling node structure

To convert the tree of Figure (A) into this representation:

1. First note that every node has at most one leftmost child
2. At most one closest right sibling.

Ex:

- In Figure (A), the leftmost child of A is B, and the leftmost child of D is H.
- The closest right sibling of B is C, and the closest right sibling of H is I.
- Choose the nodes based on how the tree is drawn. The left child field of each node points to its leftmost child (if any), and the right sibling field points to its closest right sibling (if any).

Figure (D) shows the tree of Figure (A) redrawn using the left child-right sibling representation.

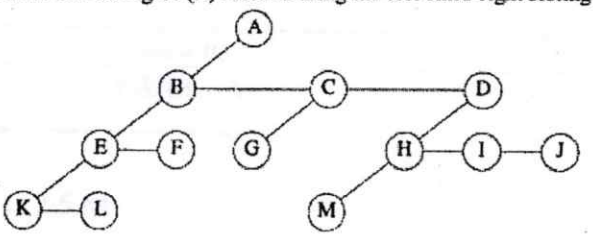


Figure (D): Left child-right sibling representation of tree of figure (A)

b

Binary tree definition – 1 M

1. Strictly Binary Tree – 1.5 M
2. Skewed Binary Tree – 1.5 M

1. Skewed Tree

A skewed tree is a tree, skewed to the left or skews to the right.

or

It is a tree consisting of only left subtree or only right subtree.

- A tree with only left subtrees is called Left Skewed Binary Tree.
- A tree with only right subtrees is called Right Skewed Binary Tree.

2. Complete Binary Tree

A binary tree T is said to complete if all its levels, except possibly the last level, have the maximum number node 2^i, $i \geq 0$ and if all the nodes at the last level appears as far left as possible.

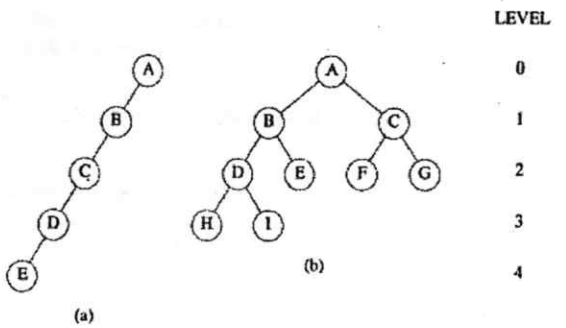


Figure (a): Skewed binary tree Figure (b): Complete binary tree

4

Modified

26

CBCS SCHEME

USN | | | | | | | | | | |

21CS32

Third Semester B.E. Degree Examination, Jan./Feb. 2023
Data Structures and Applications

Time: 3 hrs. Max. Marks: 100

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Module-1

1 a. What is linear array? Discuss the representation of linear array in memory. **(06 Marks)**

b. Differentiate between static and dynamic memory allocations. Discuss four dynamic memory allocation functions. **(06 Marks)**

c. Write a menu driven program in C for the following array operations:
(i) Inserting an element (ELEM) at a given valid position.
(ii) Deleting an element at a given valid position.
(iii) Display of array elements.
(iv) Exit
Support the program with functions for each of the above operations. **(08 Marks)**

OR

2 a. Give Abstract Data Type (ADT) for arrays. How array can be declared and initialized? **(06 Marks)**

b. With suitable example, discuss self-referential structures. **(06 Marks)**

c. Define Sparse matrix. How to represent a Sparse matrix? Write an algorithm/function to transpose a given Sparse matrix. **(08 Marks)**

Module-2

3 a. Define Stack. Discuss how to represent stack using dynamic arrays. **(06 Marks)**

b. Write a menu driven C program for the following operations on STACK of integers:
(i) Push an element on to stack
(ii) Pop an element from the stack
(iii) Display the contest of stack
(iv) Exit
Show the overflow and underflow conditions. **(06 Marks)**

c. What are the disadvantages of ordinary queue? Discuss the implementation of circular queue using arrays. **(08 Marks)**

OR

4 a. What is Recursion? Write recursive function to solve Towers of Hanoi problem. **(06 Marks)**

b. Discuss the following:
(i) Double Ended Queue
(ii) Priority Queue **(06 Marks)**

c. Write an algorithm to convert infix expression to postfix expression. Show the content of stack to convert the following infix expression:
A * (B + D)/E – F * (G + H/K) **(08 Marks)**

Module-3

5 a. Write a C function to concatenate two singly linked list. **(06 Marks)**

b. Give the structure definition for singly linked list. Write a C function to:
(i) Insert an element at the end
(ii) Delete a node at the beginning **(08 Marks)**

c. Discuss how to read a polynomial consisting of 'n' terms implemented using linked list. **(06 Marks)**

OR

6 a. Write a function to delete a node whose information field is specified in singly linked list. **(06 Marks)**

b. What is circular doubly linked list? Write a C function to perform the following operations on circular doubly linked list:
(i) Insert a node at the beginning
(ii) Delete a node from the leas **(08 Marks)**

c. Discuss how to implement stacks and queues using linked list. **(06 Marks)**

Module-4

7 a. Define binary tree. List and discuss any two properties of binary tree. **(06 Marks)**

b. Write a function to perform the following operations on Binary Search Tree (BST):
(i) Deletion from a BST
(ii) Inserting an element into a BST **(08 Marks)**

c. Define Threaded Binary Tree. Discuss In-threaded binary tree. **(06 Marks)**

OR

8 a. Discuss how binary tree are represented using (i) Array (ii) Linked list **(06 Marks)**

b. Discuss inorder, preorder, postorder and level order traversal with suitable recursive function for each. **(08 Marks)**

c. Write a C function to evaluate an expression using expression tree. **(06 Marks)**

Module-5

9 a. Design a C program for the following operation on Graph (G) of cities:
(i) Create a graph of N cities using adjacency matrix
(ii) Print all the nodes reachable from a given starting node in a digraph using BFS/DFS method **(10 Marks)**

b. Discuss AVL tree with an example. Write a function for insertion into an AVL tree. **(10 Marks)**

OR

10 a. Define hashing. What are the two criteria, a good hash function should satisfy? Discuss open addressing and chaining method with an example. **(10 Marks)**

b. Define Red-Black tree, Splay tree and B tree. Discuss the method to insert an element into Red-Black tree. **(10 Marks)**

* * * * *

Visvesvaraya Technological University
Belagavi, Karnataka - 590 018.

21CS32

Scheme & Solutions

Signature of Scrutinizer

Subject Title : Data Structures and Applications Subject Code : 21CS32

Question Number	Solution	Marks Allocated
	Module-1	
1 (a)	Linear array is a set of finite number of homogeneous elements. → (01) Representation: Show the content of memory along with address, value and name. Formula to find the address of i^{th} element → (05)	(06)
(b)	Differences between static and dynamic memory allocations → (02) Four dynamic memory allocation functions 1) malloc() 2) calloc() 3) realloc() 4) free() } Explanation required → (04)	(06)
(c)	C program includes to array operations	8
2 (a)	ADT for arrays → (03) Declaration: data type arrayname [MAXSIZE]; Initialization: data type arrayname [MAXSIZE] = {value1, value2 .. valuen};	

"APPROVED"
Registrar (Evaluation)
Visvesvaraya Technological University
BELAGAVI-590018

Visvesvaraya Technological University

Belagavi, Karnataka - 590 018.

21CS32

Scheme & Solutions

Signature of Scrutinizer

Subject Title : Data Structures and Applications Subject Code : 21CS32

Question Number	Solution	Marks Allocated
	Module-1	
1 (a)	Linear array is a set of finite number of homogeneous elements. → (01) Representation: Show the content of memory along with address, value and name. Formula to find the address of ith element → (05)	(06)
(b)	Differences between static and dynamic memory allocations → (02) Four dynamic memory allocation functions 1) malloc() 2) calloc() 3) realloc() 4) free() } Explanation required → (04)	(06)
(c)	C program includes to array operations	8
2 (a)	ADT for arrays → (03) Declaration: data type array_name [MAXSIZE]; Initialization: data type array_name [MAXSIZE] = {value1, value2 .. valuen};	

"APPROVED"

Registrar (Evaluation)
Visvesvaraya Technological University
BELAGAVI - 590018

Subject Title: Data Structures and Applications Subject Code: 21CS32

Question Number	Solution	Marks Allocated
	Give an example for both declaration and Initialization → (03)	06
(b)	Discussion on Self-Referential structure → (03) Example or simple program → (03)	06
(c)	Definition of Sparse matrix → (2) Representation of Sparse matrix → (2) Transpose of a Sparse matrix → (04)	08
	Module-2	
3 (a)	Definition of stack → (02) Using dynamic array: - Allocate or create a stack dynamically during run time - when we are not sure about the size of the stack	

```
int stacksize=1;
int *s;   int top=-1;
s=(int *) malloc (stacksize*sizeof(int));
void push()
{ if (top == stacksize-1)
{ Printf ("stack full; increase by 1\n");
  stacksize++;
  s=(int *) realloc (s, stacksize*sizeof(int));
}  s[++top]=item;
}
```

Subject Title : Data Structures and Applications Subject Code : 21CS32

Question Number	Solution	Marks Allocated

	void pop()
	{ if (top == -1)
	{ printf ("Stack underflow\n");
	return;
	}
	printf ("Item deleted = %d\n", s[top--]);
	printf ("Stack size decreased by 1\n");
	stacksize--;
	s = (int *) realloc (s, stacksize * sizeof (int));
	}
	main()
	{ ...
	}
	→ (04)
		(6)

(b)	C program for the stack functions	(6)
(c)	Disadvantages of ordinary Queue Explanation along with one suitable example. → (03) Implementation of Circular Queue → (05)	(8)
4 (a)	Recursion Definition → (01) Towers of Hanoi function. Algorithm: Tower (n, A, B, C) 1. Move n-1 disk from A to B 2. Move one disk from A to C 3. Move n-1 disk from B to C Recursive function → (5)	(6)

Subject Title : Data Structures and Applications Subject Code : 21CS32

Question Number	Solution	Marks Allocated
b)	Double ended queue - explanation about- Insertion and deletion Insert rear, Insert front, delete front, delete rear and display → (03) ii) Priority Queue. Explanation about- Ascending and descending Priority Queue. Insertion and Deletion → (03)	(06)
(c)	Algorithm to convert infix to postfix expression A*(B+D)/E-F*(G+H/K) → (04)	

Stack

Symbol	postfix string	opnd1 opnd2
A	A	A -
*	A	*
(	A	*(
B	AB	*(
+	AB	*(+
D	ABD	*(+
)	ABD+	*
/	ABD+*	/
E	ABD+*E	/
-	ABD+*E/	-
F	ABD+*E/F	-
*	ABD+*E/F	-*
(	ABD+*E/F-	-*(

Subject Title : Data Structures and Applications Subject Code : 21CS32

Question Number	Solution	Marks Allocated
	G ABD+*E/FG -*(+ ABD+*E/FG -*(+ H ABD+*E/FGH -*(+ I ABD+*E/FGH -*(+/ K ABD+*E/FGHIK) ABD+*E/FGHK -* - ABD+*E/FGHK/+*- → (4)	(8)
5(a)	Module => C function to concatenate	(6)
(b)	structure definition. → (02) C function i) Insert at end → (03) ii) Delete at beginning → (03)	(8)

(c)	Display function.
	for (i=1; i<=n; i++)
	{ printf ("Enter the term/n");
	Printf ("cf, px, py pz");
	scanf ("%d %d %d %d", &cf, &px, &py, &pz);
	head = insertRear (cf, fx, fy, fz, head);
	} → call this function
		(6)

6(a)	Function to delete a node → (6)	(6)
(b)	Definition → (02) C function for i) Insert at beginning ii) Delete a from rear 3+3 = (6)	(08)

(c) Implementation of Stack using Linked list - 3+3 (6)
Implementation of Queue using " "

7 (a) Definition → (02)
Properties: i) The max no of nodes on level $i = 2^i$ for $i \geq 0$
ii) Max. no of nodes in a binary tree of depth $K = 2^K - 1$ (6)
iii) The no of leaf nodes is equal to no of nodes of degree 2.
Any two properties — (04)

(b) Function i) addition } 4+4 (8)
ii) Evaluation }

(c) Definition. → (02) (6)
Discussion on In-Threaded binary tree → (04)

8 (a) Binary tree representation using array.
" " using Linked list 3+3=6 (6)

(b) Inorder, preorder, postorder, levelorder (8)
Explanation / Algorithm with example 2×4=8

(c) C function to evaluate the expression (6)

9 (a) C program on operations on Graph 10

(b) AVL tree definition and one example → (03)
Function for insertion → (07) (10)

"APPROVED"
Registrar (Evaluation)
Visvesvaraya Technological University
BELAGAVI - 590018

(c) Implementation of Stack using Linked list - 3+3 (6)
Implementation of Queue using " 3+3

7 (a) Definition → (02)
Properties: i) The max no of nodes on level $i = 2^i$ for $i \geq 0$
ii) max. no of nodes in a binary tree of depth $K = 2^K - 1$ (6)
iii) The no of Leaf nodes is equal to no of nodes of degree 2.
Any two properties – (04)

(b) Function i) Deletion } 4+4 (8)
ii) Inserting }

(c) Definition → (02) (6)
Diagram on In-Threaded binary tree → (04)

8 (a) Binary tree representation using array. (6)
" " using Linked list 3+3=6

(b) Inorder, preorder, postorder, levelorder (8)
Explanation / Algorithm with example 2×4=8

(c) C function to evaluate the expression (6)

9 (a) C program on operation on Graph 10

(b) AVL tree definition and one example → (03) (10)
Function for insertion → (07)

"APPROVED"
Registrar (Evaluation)
Visvesvaraya Technological University
BELAGAVI - 590018

10.(a) Definition of hashing → (02)

Two criteria:

1) A hash function should generate the hash addr such that all the keys are distributed as evenly as p among the various cells of the hash table

2) Computation of a key should be simple.

Open addressing method. → (03) → (02)

Chaining method → (03)

(b) Definition of Red-black tree
Splay tree
B tree
} (06)

Method to insert an element → (04)

92

ALVA'S INSTITUTE OF ENGINEERING&TECHNOLOGY, MOODBIDRI

(A unit of Alva's Education Foundation)

III Semester B.E. (AIML)

21CS32 – DATA STRUCTURES AND APPLICATIONS

ASSIGNMENT 1

1. Define Data Structure. Explain in the block schematic diagram different types of data structures.
2. Define static memory allocation and dynamic memory allocation .Explain dynamic malloc and freelloc with syntax and example.
3. Define polynomials with syntax and show array representation of polynomials.
4. Define union with syntax give differences b/w union and structures.
5. What is an array? Explain Traversing and Deletion operation array is collection of homogeneous elements.
6. Define Pointer. How to initialize and declare pointers with syntax and examples.

ASSIGNMENT 2

1. Define linked list and explain node representation and representation of linked list in memory.
2. Explain insertion and deletion operation of a node in a singly linked list.
3. Define doubly linked list.Explain how nodes can be inserted at the beginning and also how the nodes can be deleted at the front and end.
4. Illustrate insertion and deletion operation in circular linked list.
5. What is polynomial with C program? Explain how to add 2 polynomial expressions.
6. Difference b/w grounded header list and circular header list.
7. Explain priority queue and circular queue.

LVA'S
tion Foundation

Alva's Institute of Engineering & Technology
Shobhavana Campus, Mijar, Moodbidri, D.K – 574225
Phone: 08258-262725, Fax: 08258-262726

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

7

DATA STRUCTURES AND APPLICATIONS
SEMESTER – III
AY 2022-23 ODD

Course Code-21CS32

Faculty Name : Prof. Soundarya B C

Content beyond the syllabus

The **Virtual Lab for Data Structures** will focus on creating an environment where the student interactively explores data structures.

- Present visual animations of data structures List comprehension
- Allow students to interactively execute algorithms in these data structures
- Allow students to interactively compute the cost of using these data structures with different algorithms
- Analysis of Stack and Queue operations.
- Visualization of Linked list operations.
- Visualization of sorting algorithms.

Sindhya BC
Course Coordinator

Signature of IQAC Member(Module)

KHS 08/12/2022
Signature of IQAC Chairman(HOD)

Dept. of Artificial Intelligence & Machine Learning
Alva's Institute of Engineering and Technology
Shobhavana Campus, Mijar
Moodubidire - 574 225, D.K, Karnataka, India

DATA STRUCTURES AND APPLICATIONS
SEMESTER – III
AY 2022-23 ODD

Course Code-21CS32

Faculty Name : Prof. Soundarya B C

Content beyond the syllabus

The **Virtual Lab for Data Structures** will focus on creating an environment where the student interactively explores data structures.

- Present visual animations of data structures List comprehension
- Allow students to interactively execute algorithms in these data structures
- Allow students to interactively compute the cost of using these data structures with different algorithms
- Analysis of Stack and Queue operations.
- Visualization of Linked list operations.
- Visualization of sorting algorithms.

Sndy BC
Course Coordinator

H.O.D
Dept. of Computer Science and Design
Alva's Institute of Engg. & Technology
Mijar, Moodubidire - 574 225

Different Modes of content delivery

Lecturing (through black board, presentation and demonstration)

The most common way to convey information in the university classroom is through lectures. It might be helpful to think about the lecture as a learning experience that connects the audience, content, and lecturer. In order to maximize the best practices around lecturing there are a few things to keep in mind. First, focus on a single topic – know what your objectives for the lecture are. What three to five things do you want your students to come away from the lecture with? Second, synchronize slides (and images) to go with your verbal presentation. Select graphics that represent the ideas, concepts or words. Finally, know your lecture style and what you're comfortable with.

Flipped classroom

The flipped classroom is a pedagogical model where lecture and homework elements are reversed. This means that a recording of a lecture posted in the google can be viewed by students at home before coming to the lecture, and then the contact time itself is devoted to discussion and activities.

Problem based learning

Whereas in traditional teaching a lecturer would give students information or the 'answers', in problem based learning you present students with a problem rather than a solution. This allows students to become more active in their learning as they work out which information they need to find out to solve a particular problem. There are many advantages to students in using this approach, as it allows them to:

- Develop transferable and employability skills that will be useful in the workplace
- Improve communication and team working
- Practice research and information processing
- Develop debating and analytical skills

Peer Assisted Learning or Student led learning

Peer Assisted Learning involves students (typically from the fast learning category) facilitating discussion sessions for students from the slow or moderate learning category. Student- or peer-led learning is where students themselves facilitate their learning, often by students in the year above guiding students in group activities to discuss materials with their peers and solve problems. This helps them to think through what they have previously been taught and encourages collaborative learning.

ALVA'S INSTITUTE OF ENGINEERING & TECHNOLOGY
(Unit of Alva's Education Foundation (R), Moodbidri)
Affiliated to Visvesvaraya Technological University, Belagavi & Approved by AICTE, New Delhi. Recognized by Government of Karnataka.
Shobhavana Campus, MIJAR-574225, Moodbidri, D.K., Karnataka
Department of Computer Science and Design

DATA STRUCTURES AND APPLICATIONS
SEMESTER – III
AY 2022-23 ODD

Course Code-21CS32

Faculty Name : Prof. Soundarya B C

Modes of content delivery

Module No	Modes of content delivery
I	Lecture through black board
II	Lecture through presentation
III	Lecture through demonstration
IV	Peer Assisted learning

Course Coordinator

H.O.D
Dept. of Computer Science and Design
Alva's Institute of Engg. & Technology
Mijar, Moodubidire - 574 225

LVAS INSTITUTE OF ENGINEERING AND TECHNOLOGY MOODBIDRI

Branch : CG Semester : 3

Sl NO.	USN	21CS32
1	4AL21CG001	26 (TH) , 15 (PR)
2	4AL21CG002	22 (TH) , 14 (PR)
3	4AL21CG003	19 (TH) , 14 (PR)
4	4AL21CG004	26 (TH) , 13 (PR)
5	4AL21CG005	27 (TH) , 20 (PR)
6	4AL21CG006	23 (TH) , 12 (PR)
7	4AL21CG007	23 (TH) , 15 (PR)
8	4AL21CG008	25 (TH) , 14 (PR)
9	4AL21CG009	27 (TH) , 15 (PR)
10	4AL21CG010	29 (TH) , 16 (PR)
11	4AL21CG011	25 (TH) , 14 (PR)
12	4AL21CG012	26 (TH) , 15 (PR)
13	4AL21CG013	29 (TH) , 18 (PR)
14	4AL21CG014	26 (TH) , 13 (PR)
15	4AL21CG015	21 (TH) , 17 (PR)
16	4AL21CG016	30 (TH) , 18 (PR)
17	4AL21CG017	26 (TH) , 16 (PR)
18	4AL21CG018	28 (TH) , 15 (PR)
19	4AL21CG019	30 (TH) , 15 (PR)
20	4AL21CG020	21 (TH) , 15 (PR)
21	4AL21CG021	22 (TH) , 15 (PR)
22	4AL21CG022	25 (TH) , 17 (PR)
23	4AL21CG023	25 (TH) , 16 (PR)
24	4AL21CG024	27 (TH) , 15 (PR)
25	4AL21CG025	23 (TH) , 15 (PR)
26	4AL21CG026	25 (TH) , 14 (PR)
27	4AL21CG027	28 (TH) , 14 (PR)
28	4AL21CG028	24 (TH) , 12 (PR)
29	4AL21CG029	26 (TH) , 18 (PR)
30	4AL21CG030	20 (TH) , 15 (PR)
31	4AL21CG031	26 (TH) , 18 (PR)
32	4AL21CG032	25 (TH) , 18 (PR)
33	4AL21CG033	23 (TH) , 14 (PR)
34	4AL21CG034	19 (TH) , 14 (PR)
35	4AL21CG035	24 (TH) , 18 (PR)
36	4AL21CG036	27 (TH) , 18 (PR)

Sl NO.	USN	21CS32
37	4AL21CG037	24 (TH) , 14 (PR)
38	4AL21CG038	26 (TH) , 15 (PR)
39	4AL21CG039	30 (TH) , 20 (PR)
40	4AL21CG040	26 (TH) , 15 (PR)
41	4AL21CG041	21 (TH) , 15 (PR)
42	4AL21CG042	23 (TH) , 14 (PR)
43	4AL21CG043	26 (TH) , 14 (PR)
44	4AL21CG044	22 (TH) , 13 (PR)
45	4AL21CG045	22 (TH) , 16 (PR)
46	4AL21CG046	26 (TH) , 18 (PR)
47	4AL21CG047	27 (TH) , 17 (PR)
48	4AL21CG048	26 (TH) , 16 (PR)
49	4AL21CG049	24 (TH) , 15 (PR)
50	4AL21CG051	30 (TH) , 15 (PR)
51	4AL21CG052	27 (TH) , 18 (PR)
52	4AL21CG053	29 (TH) , 18 (PR)
53	4AL21CG054	25 (TH) , 16 (PR)
54	4AL21CG055	22 (TH) , 15 (PR)
55	4AL21CG056	27 (TH) , 12 (PR)
56	4AL21CG057	27 (TH) , 12 (PR)
57	4AL21CG058	25 (TH) , 18 (PR)
58	4AL21CG059	27 (TH) , 16 (PR)
59	4AL21CG060	27 (TH) , 14 (PR)
60	4AL21CG061	22 (TH) , 14 (PR)
61	4AL21CG062	29 (TH) , 17 (PR)
62	4AL21CG063	28 (TH) , 17 (PR)
63	4AL22CG400	20 (TH) , 18 (PR)
64	4AL22CG401	26 (TH) , 17 (PR)
65	4AL22CG402	25 (TH) , 18 (PR)