CHHATTISGARH SWAMI VIVEKANAND TECHNICAL UNIVERSITY, BHILAI

For Practical Examination Only

Session: Nov-Dec 2021 Examination

Name of Examination: B.tech, Regular **Semester:** 3rd **Date:** 05./.04./2022

Subject Code: B022(022) **Subject Name:** DSA Laboratory

Name of Student: V Om Sai Nageshwar Sharma

Roll No. of Student: 303302220020 **Enroll No:** BJ4599

Name of the Institution: SSIPMT, Raipur **Institute Code:** 033

Name of Experiment Allotted: Q1) Describe directed graph along with its weight and adjacency matrix.

Q2.) Write a pragram in C/C++ to implement binary tree using linked list.

Signature of Student

CHHATTISGARH SWAMI VIVEKANAND TECHNICAL UNIVERSITY

Bhilai, Durg, Chhattisgarh

Admit Card for Session Nov-Dec 2021

STUDENT DETAILS

Registration No.	BJ4599	Roll No.	303302220020
College Name	033 - SHRI SHANKARACHARYA INSTITUTE OF PROFESSIONAL MANAGEMENT & TECHNOLOGY,RAIPUR		
Student's Name	V OM SAI NAGESHWAR SHARMA		
Gender	Male	Date of Birth	17-02-2001
Father's Name	V Somshekhar Sharma	Current Semester	3 SEMESTER
Program	B Tech Computer Science Engineering	Course	B.Tech

SUBJECT DETAILS

Sl	Semester	Subject Type	Subject Code	Subject Name	Exam Session	Exam Type	ToAppear	Date & Time of Exam
1	3 SEMESTER	Sessional	B000308(046)	Personality Development	Nov-Dec 2021	Regular	Y	
2	3 SEMESTER	Theory	B000311(022)	Data structure & Algorithms	Nov-Dec 2021	Regular	Y	23/03/22 10:00 am
3	3 SEMESTER	Theory	B000312(014)	Mathematics - III	Nov-Dec 2021	Regular	Y	21/03/22 10:00 am
4	3 SEMESTER	Theory	B000313(022)	Principles of Programming Languages	Nov-Dec 2021	Regular	Y	25/03/22 10:00 am
5	3 SEMESTER	Theory	B000314(022)	Digital Electronics& Logic Design	Nov-Dec 2021	Regular	Y	28/03/22 10:00 am
6	3 SEMESTER	Theory	B000315(022)	Operating Systems	Nov-Dec 2021	Regular	Y	30/03/22 10:00 am
7	3 SEMESTER	Practical	B000321(022)	Data structure & Algorithms Laboratory	Nov-Dec 2021	Regular	Y	05/04/22 10:00 am
8	3 SEMESTER	Practical	B000322(022)	Digital Electronics& Logic Design Laboratory	Nov-Dec 2021	Regular	Y	
9	3 SEMESTER	Practical	B000323(022)	Operating Systems Laboratory (UNIX)	Nov-Dec 2021	Regular	Y	01/04/22 10:00 am
10	3 SEMESTER	Practical	B000324(022)	Software Laboratory (Sci Lab/MATLAB)	Nov-Dec 2021	Regular	Y	04/04/22 10:00 am

Signature should match with uploaded signature of candidate		
Signature of the Candidate (while receiving)	Signature of the Principal	Signature of COE

INSTRUCTION FOR WRITTEN EXAMINATION

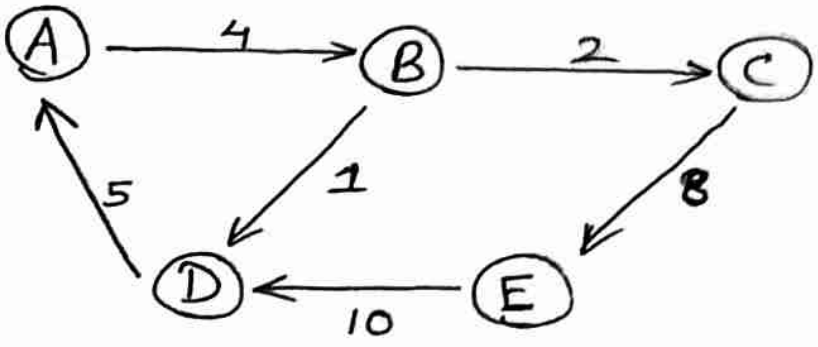
Answer 1: Weighted Directed graph

A weighted directed graph (also known as directed graphs with weights assigned to their arrows, similarly to weighted graphs (which are also known as undirected networks).

A weighted directed graph with the added feature of each edge having a value or a weight. This weight value allows for more complex problems to be expressed through graphs.

An Adjacency Matrix: It is a 2-D array of size V time V, where V is the no. of vertices in a graph. For example, if an array (M), M[i, j] = 1 indicates there is an edge from vertice i to vertex j. An adjacency matrix for an undirected graph is always symmetric. An adjacency matrix can also be used to represent weighted graphs. For example, if M(i, j) = w, then there is an edge from vertex i to vertex j with weight w.

Example :



Weighted Directed Graph

The Adjacency matrix for the above graph is

	A	B	C	D	E
A	0	4	0	0	0
B	0	0	2	1	0
C	0	0	0	0	8
D	5	0	0	0	0
E	0	0	0	10	0

Answer 2: C program to implement Binary Tree using linked list.

```
#include <stdio.h>
#include <malloc.h>

struct node {
    struct node * left;
    char data;
    struct node * right;
};

struct node * constructTree (int);
void inorder (struct node *);
char array [] = { 'A', 'B', 'C', 'D', 'E', 'F', 'G',
                  '\0', '\0', 'H' };
int leftcount [] = { 1, 3, 5, -1, 9, -1, -1, -1,
                     -1, -1, -1};
int rightcount[] = { 2, 4, 6, -1, -1, -1, -1, -1,
                     -1, -1 };
```

```
Void main() {
    struct node *root;
    root = constructTree(0);
    printf("In-order Traversal: \n");
    inorder(root);
}

struct node* constructTree(int index) {
    struct node *temp = Null;
    if (index != -1) {
        temp = (struct node *) malloc(sizeof
            (struct node));
        temp -> left = constructTree(leftcount
        [index]);
        temp -> data = array[index];
        temp -> right = constructTree(rightcount
            [index]);
        }
        return temp;
    }
void inorder(struct node* root) {
    if (root != Null) {
        inorder(root -> left);
        printf("%c\t", root -> data);
        inorder(root -> right);
    }
}
```

Output :

In-order Traversal :

D B H E A F C G.