Institute of Computer Technology

B. Tech. Computer Science and Engineering

Sub: DS

Course Code: 2CSE302

Practical – 17

Name: Jaymin Gondaliya

Enrollment No: 23162171007

Sem - 3

Branch: CS

Class: A

Batch: 32

You are exploring a maze. The maze is represented as a graph where each room is a node, and each passage between rooms is an edge. You start at the Entrance (Node A) and want to find the Treasure. However, the maze has multiple paths, and you must explore as deeply as possible before backtracking. This exploration method is Depth First Search (DFS).

Code:

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_NODES 26 // For alphabets A-Z

// Structure to represent a node in the adjacency list
typedef struct Node {
    char data;
    struct Node* next;
} Node;

// Graph structure
typedef struct Graph {
    Node* adjList[MAX_NODES];
    int visited[MAX_NODES];
```

```c
} Graph;

// Function to create a new node
Node* createNode(char data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->next = NULL;
    return newNode;
}

// Initialize the graph
void initializeGraph(Graph* graph, int numNodes) {
    for (int i = 0; i < MAX_NODES; i++) {
        graph->adjList[i] = NULL;
        graph->visited[i] = 0;
    }
}

// Add an edge to the graph
void addEdge(Graph* graph, char src, char dest) {
    int srcIdx = src - 'A';
    Node* newNode = createNode(dest);
    newNode->next = graph->adjList[srcIdx];
    graph->adjList[srcIdx] = newNode;
}

// DFS function
void DFS(Graph* graph, char start) {
    int startIdx = start - 'A';
    graph->visited[startIdx] = 1;
    printf("%c ", start);

    Node* temp = graph->adjList[startIdx];
    while (temp != NULL) {
        int neighborIdx = temp->data - 'A';
        if (!graph->visited[neighborIdx]) {
            DFS(graph, temp->data);
        }
        temp = temp->next;
    }
}

int main() {
    Graph graph;
    int numNodes, numEdges;
    char src, dest, start;

    // Input number of nodes and edges
```

```c
    printf("Enter the number of nodes: ");
    scanf("%d", &numNodes);

    printf("Enter the number of edges: ");
    scanf("%d", &numEdges);

    // Initialize the graph
    initializeGraph(&graph, numNodes);

    // Input the edges
    printf("Enter the edges (e.g., A B for an edge from A to B):\n");
    for (int i = 0; i < numEdges; i++) {
        getchar(); // Consume the newline character
        printf("Edge %d: ", i + 1);
        scanf("%c %c", &src, &dest);
        addEdge(&graph, src, dest);
    }

    // Input the starting node
    getchar(); // Consume the newline character
    printf("Enter the starting node: ");
    scanf("%c", &start);

    // Perform DFS traversal
    printf("DFS Traversal Path: ");
    DFS(&graph, start);
    printf("\n");

    return 0;
}
```

Output:

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SERIAL MONITOR  COMMENTS

PS C:\ICT\SEM-3\DS\Practical> cd 'c:\ICT\SEM-3\DS\Practical\Practical-17\output'
PS C:\ICT\SEM-3\DS\Practical\Practical-17\output> & .\'main.exe'
Enter the number of nodes: 7
Enter the number of edges: 6
Enter the edges (e.g., A B for an edge from A to B):
Edge 1: A B
Edge 2: A C
Edge 3: B D
Edge 4: C D
Edge 5: D E
Edge 6: E F
Enter the starting node: A
DFS Traversal Path: A C D E F B
PS C:\ICT\SEM-3\DS\Practical\Practical-17\output>
```