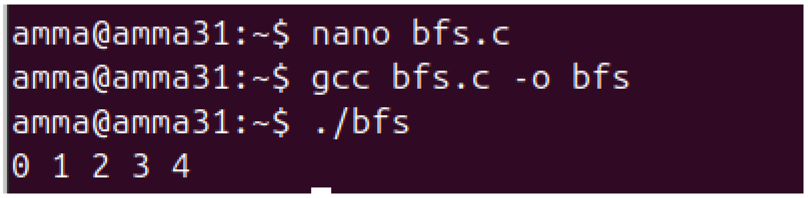
DAA ASSIGNMENT – 3

1.BFS

PROGRAM:

```
#include <stdio.h>
#define V 5
#define MAXQ 100
void bfs(int adj[V][V], int res[V], int *resSize) {
    int visited[V] = {0};
    int q[MAXQ];
    int front = 0, rear = 0;
    int src = 0;
    visited[src] = 1;
    q[rear++] = src;

    while (front < rear) {
        int curr = q[front++];
        res[(*resSize)++] = curr;
        for (int x = 0; x < V; x++) {
            if (adj[curr][x] && !visited[x]) {
                visited[x] = 1;
                q[rear++] = x;
            }
        }
    }
}

void addEdge(int adj[V][V], int u, int v) {
    adj[u][v] = 1;
    adj[v][u] = 1;  // undirected
}

int main() {
    int adj[V][V] = {0};

    addEdge(adj, 1, 2);
    addEdge(adj, 1, 0);
    addEdge(adj, 2, 0);
    addEdge(adj, 2, 3);
    addEdge(adj, 2, 4)
    int res[V];
    int resSize = 0;
    bfs(adj, res, &resSize);
    for (int i = 0; i < resSize; i++)
        printf("%d ", res[i]);
    return 0;
}
```

OUTPUT:



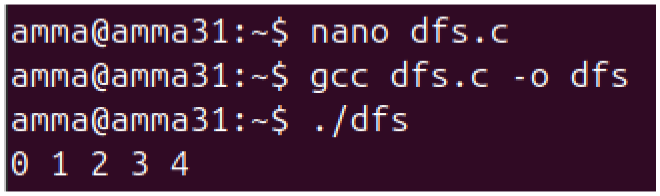
```
amma@amma31:~$ nano bfs.c
amma@amma31:~$ gcc bfs.c -o bfs
amma@amma31:~$ ./bfs
0 1 2 3 4
```

DFS:

PROGRAM:

```
#include <stdio.h>
#define V 5
void dfsRec(int adj[V][V], int visited[V], int s, int res[V], int *idx) {
    visited[s] = 1;
    res[(*idx)++] = s;

    for (int i = 0; i < V; i++) {
        if (adj[s][i] && visited[i] == 0)
            dfsRec(adj, visited, i, res, idx);
    }
}

void dfs(int adj[V][V], int res[V]) {
    int visited[V] = {0};
    int idx = 0;
    dfsRec(adj, visited, 0, res, &idx);

void addEdge(int adj[V][V], int u, int v) {
    adj[u][v] = 1;
    adj[v][u] = 1;
}

int main() {
    int adj[V][V] = {0};

    addEdge(adj, 1, 2);
    addEdge(adj, 1, 0);
    addEdge(adj, 2, 0);
    addEdge(adj, 2, 3);
    addEdge(adj, 2, 4);

    int res[V];
    dfs(adj, res);

    for (int i = 0; i < V; i++)
        printf("%d ", res[i]);

    return 0;
}
```

OUTPUT:



```
amma@amma31:~$ nano dfs.c
amma@amma31:~$ gcc dfs.c -o dfs
amma@amma31:~$ ./dfs
0 1 2 3 4
```