

- (a) Write a program to traverse a graph using DFS method
- (b) Write a program to check whether given graph is connected or not using DFS method.

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

```
int n, i, j, visited[10], queue[10], front = -1, rear = -1;
```

```
int adj[10][10];
```

```
void bfs(int v)
```

```
{
```

```
for (i = 1; i <= n; i++)
```

```
if (adj[v][i] && !visited[i])
```

```
queue[rear++] = i;
```

```
if (front <= rear)
```

```
{
```

```
visited[queue[front]] = 1;
```

```
bfs(queue[front++]);
```

```
}
```

```
}
```

```
void main()
```

```
{
```

```
int v;
```

```
printf("Enter the number of vertices");
```

```
scanf("%d", &n);
```

```
for (i = 1; i <= n; i++) {
```

```
queue[i] = 0;
```

```
visited[i] = 0;
```

```
}
```

```
printf("Enter graph data in matrix form");
```

```
for (i = 1; i <= n; i++)
```

```
for (j = 1; j <= n; j++)
```

```
scanf("%d", &adj[i][j]);
```

```

printf ("Enter the starting vertex", &v);
scanf ("%d", &v);
bfs (v);
printf ("The nodes which are reachable are:");
for (i=1; i<=n; i++)
    if (visited[i])
        printf ("%d ", i);
    else
        printf ("BFS is not possible. Not all nodes are reachable");
}

```

output → Enter the number of vertices: 4

Enter graph data in matrix form:

0 1 1 0

1 0 0 0

1 0 0 1

0 1 1 0

Enter the starting vertex: 2

The nodes which are reachable are:

2 2 3 4

~~955~~ thanked



5)

```
#include <stdio.h>
#include <conio.h>
int a[20][20], reach[20], n;
void dfs(int v) {
    int i;
    reach[v] = 1;
    for (i = 1; i <= n; i++)
        if (a[v][i] && !reach[i])
            printf("in 1.d → 1.d", v, i);
            dfs(i);
}
```

```
int main(int argc, char *args) {
```

```
    int i, j, count = 0;
    printf("Enter number of vertices");
    scanf("%d", &n);
```

```
    for (i = 1; i <= n; i++) {
        reach[i] = 0;
        for (j = 1; j <= n; j++)
            a[i][j] = 0;
```

```
    printf("Enter the adjacency matrix");
```

```
    for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++)
            scanf("%d", &a[i][j]);
```

dfs(2);

print("1m");

for (i=1; i<=n; i++) {

if (adj[i][2])

count++;

}

if (count == n)

print("1m Graph is connected");

else

print("1m Graph is not connected");

return 0;

}

Output → Enter number of vertices: 4

Enter the adjacency matrix

0 1 1 0

0 0 0 1

0 0 0 0

0 0 1 0

2 → 2

2 → 4

4 → 2

Graph is connected



int code

Merging of two Binary Tree  $\rightarrow$

output  $\rightarrow$  Case I

Input root 1 = [1, 3, 2, 5]

root 2 = [2, 1, 3, null, 4, null, 7]

Output

[3, 4, 5, 5, 4, null, 7]

Expected

[3, 4, 5, 5, 4, null, 7]

Case II  $\rightarrow$

Input  $\rightarrow$  root 1 = [1]

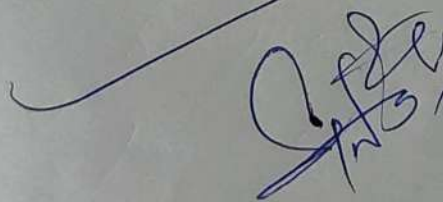
root 2 = [1, 2]

Output  $\rightarrow$

[2, 2]

Expected  $\rightarrow$

[2, 2]

 31.02.2024