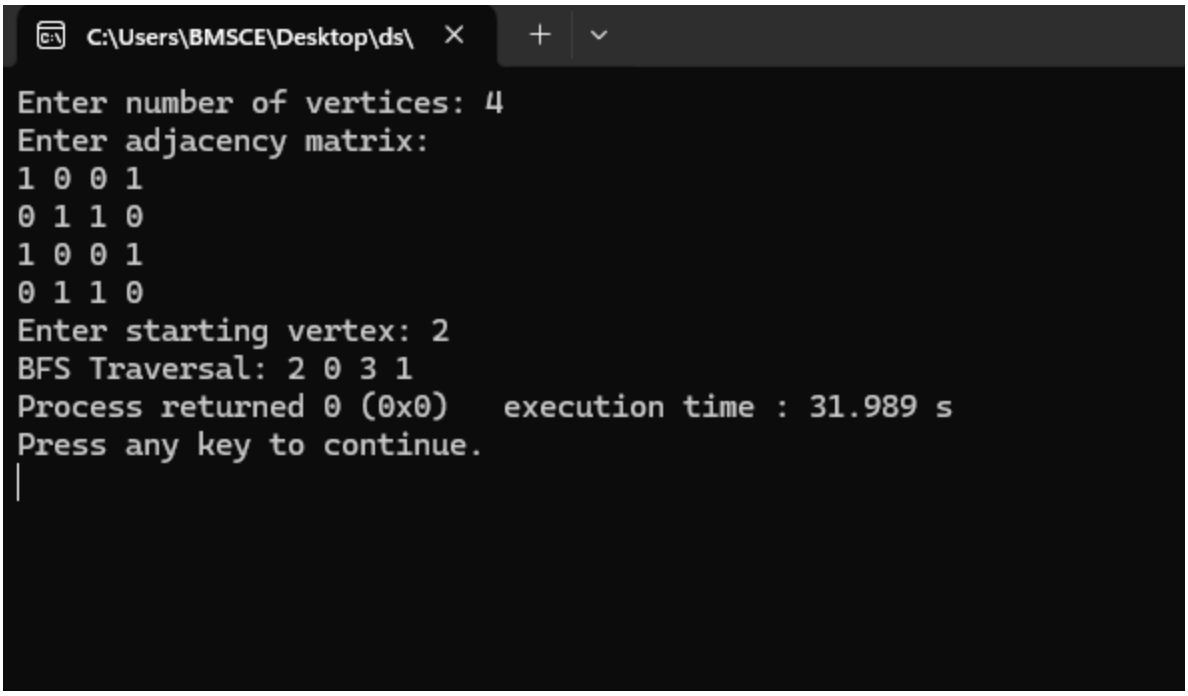
BFS CODE:

```
#include <stdio.h>

int graph[20][20], visited[20], n;

void BFS(int start) {
    int queue[20], front = 0, rear = 0;

    visited[start] = 1;
    queue[rear++] = start;

    while (front < rear) {
        int node = queue[front++];
        printf("%d ", node);

        for (int i = 0; i < n; i++) {
            if (graph[node][i] == 1 && !visited[i]) {
                visited[i] =  1;
                queue[rear++] = i;
            }
        }
    }
}

int main() {
    int start;
```

```
    printf("Enter number of vertices: ");
    scanf("%d", &n);

    printf("Enter adjacency matrix:\n");
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            scanf("%d", &graph[i][j]);

    for (int i = 0; i < n; i++)
        visited[i] = 0;

    printf("Enter starting vertex: ");
    scanf("%d", &start);

    printf("BFS Traversal: ");
    BFS(start);

    return 0;
}
```

OUTPUT:



OBSERVATION:

Page No:
Date.

22/12/25.

Prg - 9 a

WAP to traverse a graph using BFS method:

Code:

```
#include <stdio.h>
int graph[20][20], visited[20], n;

void BFS(int start) {
    int queue[20], front = 0, rear = 0;

    visited[start] = 1;
    queue[rear++] = start;

    while (front < rear) {
        int node = queue[front++];
        printf("%d", node);

        for (int i = 0; i < n; i++) {
            if (graph[node][i] == 1 && !visited[i]) {
                visited[i] = 1;
                queue[rear++] = i;
            }
        }
    }
}

int main() {
    int start;
```

```
printf ("Enter number of vertices: ");
scanf ("%d", &n);

printf ("Enter adjacency matrix: \n");
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
        scanf ("%d", &grap[i][j]);

for (int i = 0; i < n; i++)
    visited[i] = 0;

printf ("Enter starting vertex: ");
scanf ("%d", &start);

printf ("BFS Traversal: ");
BFS (start);

return 0;
}
```

Output:

```
Enter number of vertices: 4
Enter adjacency matrix:
1 0 0 1
0 1 1 0
1 0 0 1
0 1 1 0
Enter starting vertex: 2
BFS Traversal: 2 0 3 1
```