

```

#include <stdio.h>

#define MAX 20

int pushStack[MAX], popStack[MAX];
int top1 = -1, top2 = -1;
int visited[MAX];

void push1(int v) {
    pushStack[++top1] = v;
}

int pop1() {
    if (top1 == -1) return -1;
    return pushStack[top1--];
}

void push2(int v) {
    popStack[++top2] = v;
}

int pop2() {
    if (top2 == -1) return -1;
    return popStack[top2--];
}

int dequeue() {
    int v;

    if (top2 == -1) {
        while (top1 != -1) {
            push2(pop1());
        }
    }

    return pop2();
}

void BFS_Using_Stacks(int adj[MAX][MAX], int n, int start) {
    for (int i = 0; i < n; i++)
        visited[i] = 0;
}

```

Build results X Search results X Cccc X Build log X Build messages X CppCheck/Vera++ X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen info X Closed files list X Thread search X

Line	Message
	== Build file: "no target" in "no project" (compiler: unknown) == == Build finished: 0 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ==

C/C++ Windows (CR+LF) WINDOWS-1252 Line 16, Col 2, Pos 256 Insert Read/Write default 11:34:36 08-12-2025 ENG IN

```

push1(start);
visited[start] = 1;

while (top1 != -1 || top2 != -1) {
    int v = dequeue();
    printf("%d ", v);

    for (int i = 0; i < n; i++) {
        if (adj[v][i] == 1 && !visited[i]) {
            push1(i);
            visited[i] = 1;
        }
    }
}

int main() {
    int n, adj[MAX][MAX];

    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", &adj[i][j]);

    printf("BFS Traversal: ");
    BFS_Using_Stacks(adj, n, 0);

    return 0;
}

```



```
C:\Users\BMSCECSE-L4\Docu X + ▾ - □ ×  
Enter number of vertices: 4  
Enter adjacency matrix:  
1 1 0 1  
0 1 1 0  
1 0 0 0  
0 1 1 1  
BFS Traversal: 0 1 3 2  
Process returned 0 (0x0) execution time : 35.632 s  
Press any key to continue.
```



ENG  
IN

11:36:49  
08-12-2025

```

#include <stdio.h>

#define MAX 20

int stack[MAX], top = -1;
int visited[MAX];

void push(int v) {
    stack[++top] = v;
}

int pop() {
    if (top == -1) return -1;
    return stack[top--];
}

void DFS_Stack(int adj[MAX][MAX], int n, int start) {
    int i, v;

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

    push(start);

    while (top != -1) {
        v = pop();

        if (!visited[v]) {
            visited[v] = 1;

            for (i = n - 1; i >= 0; i--) {
                if (adj[v][i] == 1 && !visited[i])
                    push(i);
            }
        }
    }
}

```

ks X Search results X Cccc X Build log X Build messages X CppCheck/Vera++ X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen info X Closed files list X Thread search X

Line	Message
	== Build file: "no target" in "no project" (compiler: unknown) == == Build finished: 0 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ==

C/C++ Windows (CR+LF) WINDOWS-1252 Line 12, Col 12, Pos 147 Insert Read/Write default 11:33:07 08-12-2025 ENG IN ABC

Search

```
int main() {
    int n, adj[MAX][MAX];
    int i, j;

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

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

    DFS_Stack(adj, n, 0);
    int connected = 1;
    for (i = 0; i < n; i++) {
        if (!visited[i]) {
            connected = 0;
            break;
        }
    }

    if (connected)
        printf("Graph is Connected\n");
    else
        printf("Graph is NOT Connected\n");

    return 0;
}
```

cks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ X CppCheck/Vera++ messages X Cscope X Debugger X DoxyBlocks X Fortran info X Closed files list X Thread search X

Line	Message
	==== Build file: "no target" in "no project" (compiler: unknown) ==== ==== Build finished: 0 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===

C/C++ Windows (CR+LF) WINDOWS-1252 Line 12, Col 12, Pos 147 Insert Read/Write default

11:33:48 08-12-2025

C:\Users\BMSCECSE-L4\Docu X + ▾

Enter number of vertices: 4

Enter adjacency matrix:

```
0 1 0 1  
1 0 1 0  
1 1 1 0  
0 0 0 1
```

Graph is Connected

Process returned 0 (0x0) execution time : 30.872 s

Press any key to continue.



ENG  
IN



11:31:16  
08-12-2025