```
void bfs(int a[10][10], int n, int u)
{
        int f, r, q[10], v;
        int s[10] = {0}; /* initialize all elements in s to 0 i.e, no node is visited */
        printf("The nodes visited from %d : ", u);
        f = 0, r = -1; // queue is empty
        q[++r] = u; // Insert u into queue
        s[u] = 1; // insert u to s
        printf("%d ", u); // print the node visited
        while ( f <= r )
        {
                u = q[f++]; // delete an element from q
                for (v = 0; v < n; v++)
                {
                        if (a[u][v] == 1) // If v is adjacent to u
                        {
                                if (s[v] == 0) // If v is not in S i.e., v has not been visited
                                {
                                        printf("%d ", v); // print the node visited
                                        s[v] = 1; // add v to s, mark it as visited
                                        q[++r] = v; // Insert v into queue
                                }
                        }
                }
        }
        printf("\n");
}
```

```
void main()
{
	int n, a[10][10], source, i, j;
	printf(“Enter the number of nodes : “);
	scanf(“%d”, &n);
	printf(“Enter the adjacency matrix:\n”);
	for (i = 0; i < n; i++)
	{
		for (j = 0; j < n; j++) scanf(“%d”, &a[i][j]);
	}
	for (source = 0; source < n; source++)
		bfs(a, n, source);
}
```

```
void dfs(int u)
{
	int v;
	s[u] = 1;
	printf("%d ", u);
	for (v = 0; v < n; v++)
	{
		if (a[u][v] == 1 && s[v] == 0) dfs(v);
	}
}


void read_adjacency_matrix(int a[10][10], int n)
{
	int i, j;
	for (i = 0; i < n; i++)
	{
		for (j = 0; j < n; j++)
		{
			scanf(“%d”, &a[i][j]);
		}
	}
}



#include <stdio.h>
```

```
int a[10][10], s[10], n; // Global variables
void main()
{
	int i, source;
	printf("Enter the number of nodes in the graph : ");
	scanf("%d", &n);
	printf("Enter the adjacency matrix:\n");
	read_adjacency_matrix(a, n);
	for (source = 0; source < n; source++)
	{
		for (i = 0; i < n; i++) s[i] = 0;
		printf("\nThe nodes reachable from %d: ", source);
		dfs(source);
	}
}
```