Write a program to implement Breadth First Search.

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

#define MAX 100

int QUEUE[MAX];
int front = -1, rear = -1;

void enqueue(int vertex)
{
	if (rear == MAX - 1)
		printf("Queue Overflow\n");
	else
	{
		if (front == -1)
		front = 0;
		rear++;
		QUEUE[rear] = vertex;
	}
}

int dequeue()
{
	int vertex;
	if (front == -1 || front > rear)
	{
		printf("Queue Underflow\n");
		return -1;
	}
	else
	{
		vertex = QUEUE[front];
		front++;
		if (front > rear)
		front = rear = -1;
		return vertex;
	}
}
```

```
void BFS(int graph[MAX][MAX], int startVertex, int totalVert)
{
	int visited[MAX], currVertex, i;
	visited[startVertex] = 1;
	enqueue(startVertex);

	printf("BFS Traversal: ");
	while (front != -1)
	{
		currVertex = dequeue();
		printf("%d ", currVertex);
		for (i = 1; i <= totalVert; i++)
		{
			if (graph[currVertex][i] == 1 && !visited[i])
			{
				visited[i] = 1;
				enqueue(i);
		}
		}
	}
	printf("\n");
}

int main()
{
	int totalVert, i, j, startVertex;
	int graph[MAX][MAX];
	printf("Enter the total number of vertices: ");
	scanf("%d", &totalVert);
	printf("Enter the %dX%d adjacency matrix of the
graph:\n",totalVert,totalVert);
	for (i = 1; i <= totalVert; i++)
	{
		for (j = 1; j <= totalVert; j++)
		scanf("%d", &graph[i][j]);
	}
	printf("Enter the starting vertex: ");
	scanf("%d", &startVertex);
	BFS(graph, startVertex, totalVert);
	return 0;
}
```

OUTPUT

```
Enter the total number of vertices: 4
Enter the 4X4 adjacency matrix of the graph:
0  1  1  1
1  0  1  1
1  1  0  1
1  1  1  0
Enter the starting vertex: 2
BFS Traversal: 2 1 3 4
```

Write a program to implement Depth First Search.

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 100

int graph[MAX][MAX], visited[MAX], currVertex;

void DFS(int startVertex, int totalVert)
{
	int stack[MAX], top = -1;
	stack[++top] = startVertex;
	visited[startVertex] = 1;
	printf("DFS Traversal: ");

	while (top != -1)
	{
		currVertex = stack[top--];
		printf("%d ", currVertex);
		for (int i = totalVert; i >= 1; i--)
		{
			if (graph[currVertex][i] == 1 && !visited[i])
			{
				stack[++top] = i;
				visited[i] = 1;
			}
		}
	}
	printf("\n");
}

int main()
{
	int totalVert, startVertex, i, j;
	printf("Enter the total number of vertices: ");
	scanf("%d", &totalVert);
	printf("Enter the %dX%d adjacency matrix of the
graph:\n",totalVert,totalVert );
	for (i = 1; i <= totalVert; i++)
	{
		for (j = 1; j <= totalVert; j++)
```

```
                scanf("%d", &graph[i][j]);
        }
        for (i = 1; i <= totalVert; i++)
                visited[i] = 0;
        printf("Enter the starting vertex: ");
        scanf("%d", &startVertex);
        DFS(startVertex, totalVert);
        return 0;
}
```

OUTPUT

```
Enter the total number of vertices: 5
Enter the 5X5 adjacency matrix of the graph:
0  1  1  0  0
0  0  0  1  1
0  0  0  0  0
0  0  0  0  0
0  0  0  0  0
Enter the starting vertex: 1
DFS Traversal: 1 2 4 5 3
```