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

#define initial 0
#define visited 1
#define MAX 100

int n;
int adj[MAX][MAX];
int state[MAX];
int stack[MAX];
int top=-1;

void DF_traversal();
void DFS(int v);

void push(int v);
void create_graph();
int pop();
int is_empty_stack();

void create_graph()
{
    int i,max_edges,origin,destin;
    printf("Enter the number of nodes:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        for(int j=0;j<n;j++)
        {
            adj[i][j]=0;
        }
    }
    max_edges=n*(n-1);
    for(i=1;i<=max_edges;i++)
    {
        printf("\nEnter edge %d (-1 -1) to quit:",i);
        scanf("%d %d",&origin,&destin);
        if(origin==-1&&destin==-1)
            break;
        if(origin>=n||destin>=n||origin<0||destin<0)
        {
            printf("\nInvalid Edge");
            i--;
        }
        else
        {
            adj[origin][destin]=1;
        }
    }
}

void DF_traversal()
{
    int v;
    for(v=0;v<n;v++){
        state[v]=initial;
    }
        printf("\nEnter starting node depth first search : ");
        scanf("%d",&v);
        DFS(v);
        printf("\n");
}

void DFS(int v){
    int i;
```

```
    push(v);
    while(!is_empty_stack()){
        v=pop();
        if (state[v]==initial){
            printf("%d ", v);
            state[v] = visited;
        }
        for (i=n-1;i>=0;i--) {
            if (adj[v][i]==1&&state[i]==initial)
                push(i);
        }
    }
}


void push(int v)
{
    if(top==MAX-1){
            printf("\nStack is Full");
    }
    else{
        top=top+1;
        stack[top]=v;
    }
}

int pop()
{
    int v;
    if(top==-1){
        printf("\nstack is empty");
    exit(0);
    }
    else{
        v=stack[top];
        top=top-1;
        return v;
    }
}

int is_empty_stack()
{
    if(top==-1)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}
int main()
{
    create_graph();
    DF_traversal();
    return 0;
}
```