1. 1) Write a program to check the connectivity of a graph using DFS (Recursion).Also compute the number of components of a graph.

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<stdbool.h>**

**int adj[5][5];**

**void dfs(int u, bool visited[]) {**

**visited[u] = true;**

**for(int v = 0; v<5; v++) {**

**if(adj[u][v]) {**

**if(!visited[v])**

**dfs(v, visited);**

**}**

**}**

**}**

**bool connected() {**

**bool \*vis = malloc(sizeof(bool[5]));**

**for(int u; u < 5; u++) {**

**for(int i = 0; i<5; i++)**

**vis[i] = false;**

**dfs(u, vis);**

**for(int i = 0; i<5; i++) {**

**if(!vis[i])**

**return false;**

**}**

**}**

**return true;**

**}**

**int main() {**

**int i,j;**

**printf("\n Enter graph data :\n");**

**for (i=0;i<5;i++)**

**for (j=0;j<5;j++)**

**scanf("%d",&adj[i][j]);**

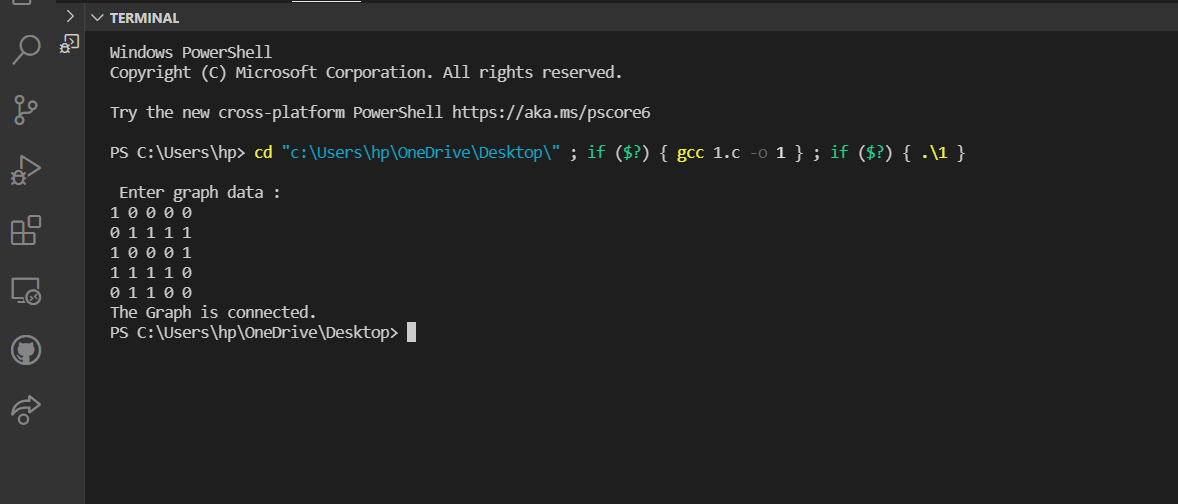
**if(connected())**

**printf("The Graph is connected.");**

**else**

**printf("The Graph is not connected.");**

**}**



**2. Write a program to traverse the graph using BFS traversal technique. Use an explicit**

**queue for implementation.**

**use adjacency matrix representation for the graph**

**#include<stdio.h>**

**int adj[50][50],q[20],vis[20],n,f=-1,r=-1;**

**void bfs(int v)**

**{**

**int i;**

**for (i=0;i<n;i++)**

**{**

**if(adj[v][i] != 0 && vis[i] == 0)**

**{**

**r=r+1;**

**q[r]=i;**

**vis[i]=1;**

**printf("%d ",i);**

**}**

**}**

**f=f+1;**

**if(f<=r)**

**bfs(q[f]);**

**}**

**int main()**

**{**

**int v,i,j;**

**printf("\n Enter the number of vertices:");**

**scanf("%d",&n);**

**for (i=0;i<n;i++)**

**{**

**vis[i]=0;**

**}**

**printf("\n Enter graph data :\n");**

**for (i=0;i<n;i++)**

**for (j=0;j<n;j++)**

**scanf("%d",&adj[i][j]);**

**printf("\n Enter the starting vertex:");**

**scanf("%d",&v);**

**f=r=0;**

**q[r]=v;**

**printf("\n BFS is:\n");**

**vis[v]=1;**

**printf(" %d ",v);**

**bfs(v);**

**if(r != n-1)**

**printf("\n BFS is not possible");**

**}**

