**Adj.Matrix and BFS:**

Code:-

#include<iostream>

#define MAX 100

using namespace std;

class nick

{

public:

int s=0;

int a[MAX][MAX];

int bfs[MAX];

int t[MAX];

int front=0;

int rear=0;

public:

void adj()

{

cout<<"how many nodes should be there? : ";

cin>>s;

cout<<s<<" by "<<s<<" Matrix ";

cout<<"\nEnter 1 if edge exists and 0 if it doesn't"<<endl<<endl;

for(int i=0;i<s;++i)

{

for(int j=0;j<s;++j)

{

cout<<endl<<"connection between node"<< i << " and node " << j <<" : ";

cin>>a[i][j];

}

}

for(int i=0;i<s;i++)

{

for(int j=0;j<s;j++)

{

cout<<"\nFor edge between node "<< i << " and node " << j <<" : ";

cout<<a[i][j];

}

}

cout<<"\nThe adjacency matrix is :";

for (int i=0;i<s;i++)

{

cout<<endl;

for (int j=0;j<s;j++)

{

cout << a[i][j]<<"\t";

}

}

}

int visited(int node)

{

for (int i=0;i<s;i++)

{

if (node==bfs[i])

{

return 0;

}

}

for (int i=front;i<rear;i++)

{

if (node==t[i])

{

return 0;

}

}

return 1;

}

void bfsfun()

{

t[0] = 0;

int c = 0;

rear=1;

int v=0;

for (int i=0;i<s;i++)

{

for (int j=0;j<s;j++)

{

if(a[v][j]==1)

{

if(visited(j) == 1)

{

t[rear] = j;

rear++;

}

}

}

bfs[c] = t[front];

c++;

front++;

v=t[front];

for (int j = 0; j <s; j++)

{

cout <<t[j]<<"\t";

}

}

bfs\_pass();

}

void bfs\_pass()

{

cout << endl;

for (int j=0;j<s;j++)

{

cout <<bfs[j]<<"\t";

}

}

};

int main()

{

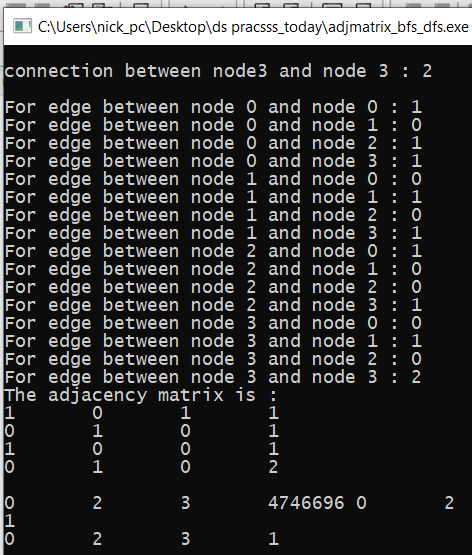
nick o;

o.adj();

o.bfsfun();

}

Output:-



**Adj.Matrix and DFS:**

Code:-

#include <iostream>

using namespace std;

class nick

{

public:

int n, a[100][100], top = 0;

char ch = 'A', c1, c2, stack[100], dfs[100];

nick()

{

mytech();

}

void mytech()

{

cout <<"Enter size (no. of n) : ";

cin >> n;

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

{

dfs[i] = '0';

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

{

c1 = ch + i;

c2 = ch + j;

cout << "Enter for " << c1 << " and " << c2 << " : ";

cin >> a[i][j];

}

}

dis();

}

void dis()

{

cout << " Adjacent Matrix : " << endl;

cout << " ";

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

{

c1 = ch + i;

cout << c1 << " ";

}

cout << endl;

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

{

c1 = ch + i;

cout << c1 << " ";

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

{

cout << a[i][j] << " ";

}

cout << endl;

}

Pass();

}

void Pass()

{

int c = 0, s = 0, count = 0;

stack[0] = 'A';

top = 0;

display\_stack();

while (s < n)

{

dfs[count] = stack[top];

count++;

c = stack[top] - 65;

top--;

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

{

if (a[c][i] == 1)

{

if (check(i) == true)

{

c1 = 65 + i;

top++;

stack[top] = c1;

}

}

}

display\_stack();

s++;

}

display\_dfs();

}

bool check(int cha)

{

char c;

c = 65 + cha;

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

{

if (c == dfs[i])

{

return false;

}

}

for (int i = 0; i <= top; i++)

{

if (c == stack[i])

{

return false;

}

}

return true;

}

void display\_dfs()

{

cout << endl << "DFS sequence : ";

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

cout << dfs[i] << " ";

cout << endl;

}

void display\_stack()

{

for (int i = 0; i <= top; i++)

cout << stack[i] << " ";

}

};

int main()

{

nick a;

}

Output:-

