```cpp
//*****************************************************************//
//*************string  Author = "SACHIN SAINI " *****************//
//*****************************************************************//


//BFS using adjacency matrix
#include<iostream>
using namespace std;
int
mat[20][20],VisitedArray[20],QueueForBFS[20],frontQueue=0,rearQueue=0,numOfVertices=0;

void BFS(int src) //src is on which we are calling BFS i.e. Which node we are expending
{
    VisitedArray[src]=1; //Visited array of source Node = 1
    QueueForBFS[rearQueue++]=src;  //Insert source node by the rear of the Queue
    while(rearQueue!=numOfVertices) //When rearQueue reaches end of the node then stop
    {
        for(int i=0;i<numOfVertices;i++) //Visit all node from one source
        {
            if(mat[src][i]==1&&VisitedArray[i]==0)
            {
                QueueForBFS[rearQueue++]=i;
                VisitedArray[i]=1;
            }
        }
        src=QueueForBFS[frontQueue];//Change the source node to front of queue
        frontQueue++; //Change the front by increasing one as previous one is visited
    }
}
void print(int n)
{
    for(int i=0;i<n;i++)
    {
        cout<<QueueForBFS[i]<<" ";
    }
}
int main()
{
    //Filling  data by own we can take from user as well
    numOfVertices=10;
    mat[0][1]=1;
    mat[1][0]=1;
    mat[0][3]=1;
    mat[3][0]=1;
    mat[3][2]=1;
    mat[2][3]=1;
    mat[1][2]=1;
    mat[2][1]=1;
    mat[1][4]=1;
    mat[4][1]=1;
    mat[1][7]=1;
    mat[7][1]=1;
    mat[6][1]=1;
    mat[1][6]=1;
    mat[4][5]=1;
    mat[5][4]=1;
    mat[4][7]=1;
    mat[7][4]=1;
    mat[4][6]=1;
    mat[6][4]=1;
    mat[6][7]=1;
    mat[7][6]=1;
    mat[2][9]=1;
    mat[9][2]=1;
    mat[2][8]=1;
    mat[8][2]=1;
    BFS(0);//Calling BFS by source 0
    print(10);
}
```

```

/////////////////////////////////////////////////////////////////////////////////////////////////////////////
Output:
0 1 3 2 4 6 7 8 9 5





```