```cpp
#include <bits/stdc++.h>
using namespace std;
#define ll long long
#define inf (int)1e9

```

## Normal Bfs:

```cpp
vector<int> D(100100 , inf) , adj[100100];

void bfs(int src){
    queue <int> q;
    q.push(src);
    D[src] = 0;
    while(!q.empty()){
        int u = q.front();
        q.pop();
        for(auto v : adj[u])
            if(D[v] == inf){
                D[v] = D[u]+1;
                q.push(v);
            }
    }
}

```

## Bfs for grid:

```cpp
vector<vector<int>> D(1000 , vector<int> (1000 , inf));
int n , m;

//for four direction:   S    E     N     w
            int dx[] = {1 , 0 , -1 ,  0};
            int dy[] = {0 , 1 ,  0 , -1};
//for eight direction:  S   SE  E    NE    N    NW   W    SW
//          int dx[] = {1 , 1 , 0 , -1 , -1 , -1 ,  0 ,  1};
//          int dy[] = {0 , 1 , 1 ,  1 ,  0 , -1 , -1 , -1};

void bfs(int x , int y){
    queue<pair<int , int>> q;
    q.push({x , y});
    D[x][y] = 0;
    while(!q.empty()){
        auto u = q.front();
        q.pop();
        for (int i = 0; i < 4; i++) { //check 8/4
            int xx = u.first+dx[i];
            int yy = u.second+dy[i];
            if(xx>=0 && xx<n && yy>=0 && yy<m && D[xx][yy]==inf){
                D[xx][yy] = D[u.first][u.second]+1;
                q.push({xx , yy});
            }
        }
    }
}
```