

**UNIVERSITY INSTITUTE OF ENGINEERING**

**Department of Computer Science & Engineering**

**Subject Name:** Competitive coding-I

**Subject Code:** 20CSP-314

|  |  |
| --- | --- |
| **Submitted By:** Kanishk Soni **Submitted To: Mr. Syed Abdul Basit** | |
| **Subject Name** | CC LAB |
| **Subject Code** | 20CSP-314 |
| **Branch** | Computer Science |
| **Semester** | 5th |

**Practical Evaluation Sheet**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr.No** | **Program** | **Date** | **Evaluation** | | | | **Sign** |
| **Conduct**  **(12)** | **Viva(8)** | **Worksheet(10)** | **Total (30)** |
| 1 | Concept of Arrays |  |  |  |  |  |  |
| 2 | Concept of Stack and Queues |  |  |  |  |  |  |
| 3 | Concept of Linked List |  |  |  |  |  |  |
| 4 | Concept of Searching and Sorting |  |  |  |  |  |  |
| 5 | Concept of Graph |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |

**Experiment 5**

**Student Name: Kanishk Soni UID: 20BCS9398**

**Branch: CSE Section/Group:707\_WM\_B**

**Semester: 5th Date of Performance: 17/10/2022**

**Subject Name: Competitive Coding - I Subject Code: 20CSP-314**

**Question1: Journey-to-the-moon**

**Code:**

#include <bits/stdc++.h>

using namespace std;

bool visited[100001] = {0};

struct node {

vector<long long> neighbour;

};

long long bfs(long long, node \*);

int main() {

long long n,m;

cin>>n>>m;

node nodelist[n];

long long a,b;

while(m--) {

cin>>a>>b;

nodelist[a].neighbour.push\_back(b);

nodelist[b].neighbour.push\_back(a);

}

long long connected = 0; //no of connected components

long long total = 0;

long long temp = 0;

vector<int> count;

for (long long i = 0; i < n; i++) {

if(!visited[i]) {

temp = bfs(i, nodelist);

count.push\_back( temp );

total += temp;

connected++;

}

}

long long answer = (total \* (total - 1)) / 2;

for (int i = 0; i < connected; i++) {

answer -= (count[i] \* (count[i] - 1)) / 2;

}

cout<<answer;

}

long long bfs(long long nod, node \*nodelist) {

int count = 0;

queue<long long> Q;

Q.push(nod);

long long n;

while(!Q.empty()) {

n = Q.front();

Q.pop();

if(visited[n]) {

continue;

}

visited[n] = true;

count++;

for (vector<long long>::iterator itr = nodelist[n].neighbour.begin(); itr != nodelist[n].neighbour.end(); itr++) {

if(!visited[\*itr]) {

Q.push(\*itr);

}

}

}

return count;

}



**Question 2:** **Frog-in-maze**

**Code:**

#include <bits/stdc++.h>

#define double long double

using namespace std;

const int MAXN = (42);

const double eps = 1e-12;

vector<double> gauss(vector<vector<double>> &a)

{

int n = a.size(), m = a[0].size() - 1;

vector<int> where(m, -1);

for(int col = 0, row = 0; col < m && row < n; col++)

{

int sel = row;

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

if(abs(a[i][col]) > abs(a[sel][col]))

sel = i;

if(abs(a[sel][col]) < eps) { where[col] = -1; continue; }

for(int i = col; i <= m; i++)

swap(a[sel][i], a[row][i]);

where[col] = row;

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

if(i != row)

{

if(abs(a[i][col]) < eps) continue;

double c = a[i][col] / a[row][col];

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

a[i][j] -= c \* a[row][j];

}

row++;

}

vector<double> ans(m, 0);

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

if(where[i] != -1)

ans[i] = a[where[i]][m] / a[where[i]][i];

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

{

double sum = a[i][m];

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

sum -= ans[j] \* a[i][j];

if(abs(sum) > eps) return vector<double>();

}

return ans;

}

int n, m, k;

string a[MAXN];

int nxt\_x[MAXN][MAXN], nxt\_y[MAXN][MAXN];

void read()

{

cin >> n >> m >> k;

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

cin >> a[i];

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

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

nxt\_x[i][j] = i, nxt\_y[i][j] = j;

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

{

int x1, y1, x2, y2;

cin >> x1 >> y1 >> x2 >> y2;

x1--; y1--; x2--; y2--;

nxt\_x[x1][y1] = x2; nxt\_y[x1][y1] = y2;

nxt\_x[x2][y2] = x1; nxt\_y[x2][y2] = y1;

}

}

int N;

int encode(int x, int y) { return x \* m + y; }

int dirx[4] = {0, 0, 1, -1};

int diry[4] = {1, -1, 0, 0};

bool ok(int x, int y)

{

if(x >= n || y >= m || x < 0 || y < 0) return false;

return a[x][y] != '#';

}

void solve()

{

N = n \* m;

vector<vector<double> > matr;

vector<double> zero(N + 1, 0);

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

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

{

if(a[i][j] == '#') { matr.push\_back(zero); continue; }

else if(a[i][j] == '\*') { matr.push\_back(zero), matr[matr.size() - 1][encode(i, j)] = 1; continue; }

else if(a[i][j] == '%') { matr.push\_back(zero), matr[matr.size() - 1][encode(i, j)] = 1; matr[matr.size() - 1][N] = 1; continue; }

vector<int> adj;

for(int d = 0; d < 4; d++)

if(ok(i + dirx[d], j + diry[d]))

adj.push\_back(encode(nxt\_x[i + dirx[d]][j + diry[d]], nxt\_y[i + dirx[d]][j + diry[d]]));

matr.push\_back(zero);

matr[matr.size() - 1][encode(i, j)] = 1;

for(int v: adj)

matr[matr.size() - 1][v] = -((double)1 / (double)adj.size());

}

vector<double> ans = gauss(matr);

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

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

if(a[i][j] == 'A')

{

cout << setprecision(9) << fixed << ans[encode(i, j)] << endl;

return;

}

}

int main()

{

ios\_base::sync\_with\_stdio(false);

cin.tie(NULL);

read();

solve();

return 0;

}

