CHANDIGARH UNIVERSITY
UNIVERSITY INSTITUTE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Submitted By: Satyam **Submitted To: Navneet Chaudhry**	
Subject Name	Competitive Coding
Subject Code	21 CSP-314
Branch	BE-CSE
Semester	5th

LAB INDEX

NAME: Satyam
SUBJECT NAME: Competitive Coding Lab
UID: 20BCS9393
SUBJECT CODE: 21CSP-314
SECTION: 607 A

Sr. No	Program	Date	Evaluation				Sign
			LW (12)	VV (8)	FW (10)	Total (30)	
1.	**ARRAYS:** https://www.hackerrank.com/challenges/30-arrays/problem https://www.hackerrank.com/challenges/simple-array-sum/problem?isFullScreen=true https://www.hackerrank.com/challenges/compare-the-triplets/problem?isFullScreen=true https://www.hackerrank.com/challenges/diagonal-difference/problem?isFullScreen=true	04-Aug-2022					
2.	**STACK & QUEUES:** https://www.hackerrank.com/challenges/equal-stacks/problem?isFullScreen=true https://www.hackerrank.com/challenges/game-of-two-stacks/problem?isFullScreen=true https://www.hackerrank.com/challenges/balanced-brackets/problem?isFullScreen=true https://www.hackerrank.com/challenges/down-to-zero-ii/problem?isFullScreen=true https://www.hackerrank.com/challenges/truck-tour/problem?isFullScreen=true	18-Aug-2022					
3.	**Linked List:** https://www.hackerrank.com/challenges/compare-two-linked-lists/problem?isFullScreen=true https://www.hackerrank.com/challenges/insert-a-node-into-a-sorted-doubly-linked-list/problem?isFullScreen=true	25-Aug-2022					

4.	**Searching and Sorting:** https://www.hackerrank.com/challenges/missing-numbers/problem?isFullScreen=true https://www.hackerrank.com/challenges/closest-numbers/problem?isFullScreen=true	01-Sep-2022					
5.	**Tree Data Structure:** https://www.hackerrank.com/challenges/bfsshortreach/problem?isFullScreen=true https://www.hackerrank.com/challenges/the-quickest-way-up/problem?isFullScreen=true https://www.hackerrank.com/challenges/even-tree/problem?isFullScreen=true https://www.hackerrank.com/challenges/journey-to-the-moon/problem?isFullScreen=true https://www.hackerrank.com/challenges/frog-in-maze/problem?isFullScreen=true	29-Sep-2022					

EXPERIMENT-2.1(a)

1. Aim/Overview of the practical:

To demonstrate the concept of Tree Data Structures.

2. Task to be done/ Which logistics used:

https://www.hackerrank.com/challenges/bfsshortreach/problem?isFullScreen=true

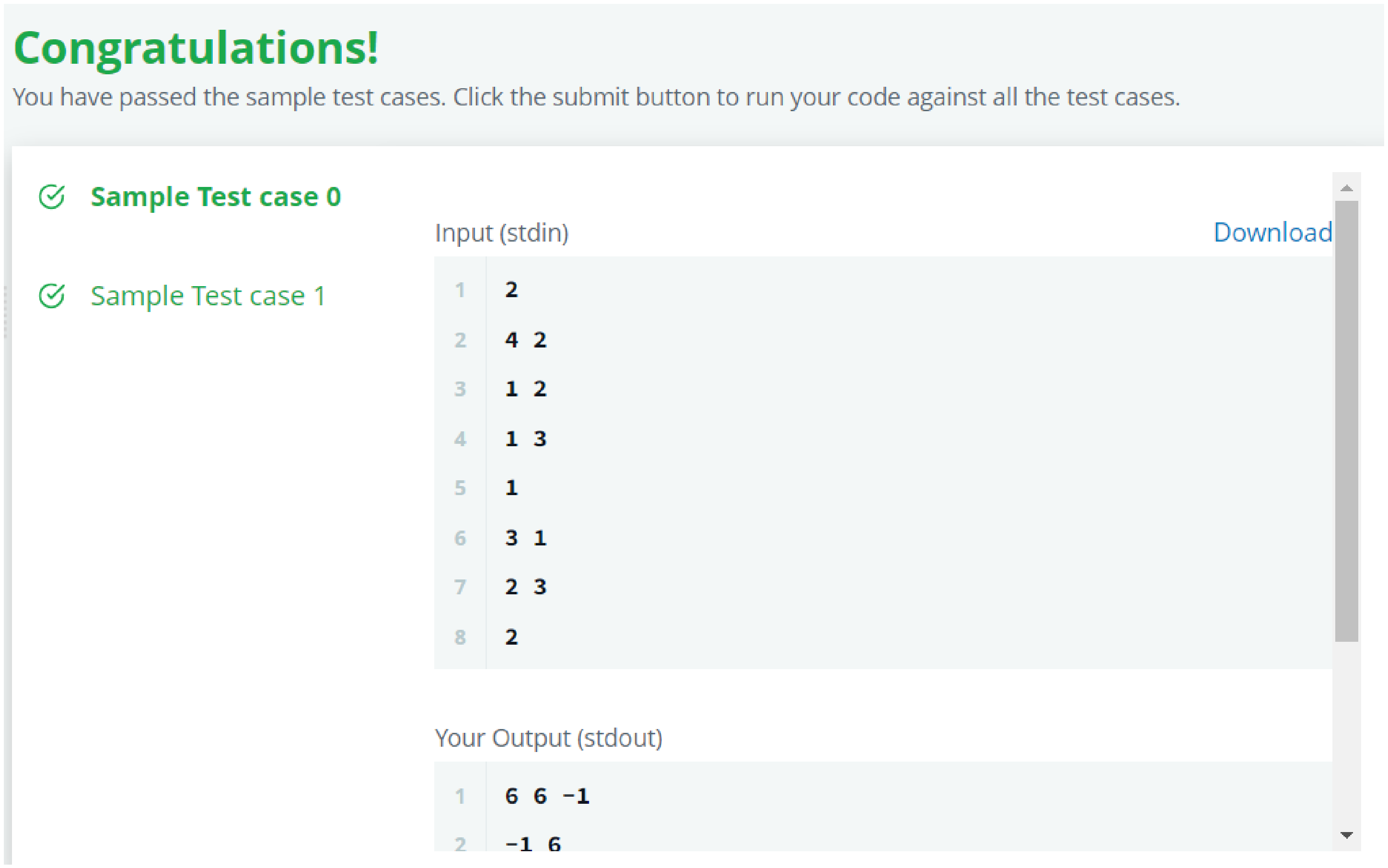
3. Steps for experiment/practical/Code:

```
#include <bits/stdc++.h>
using namespace std;
int main() {
   int t;
   cin>> t;
   while(t--)
   {
       int nodes,edges;
       cin >> nodes >> edges ;
       int mat[edges][2];
       for (int i=0;i<edges;i++)
       {
           for (int j=0;j<2;j++)
           {
               cin >> mat[i][j];
           }
       }
       int src;
       cin >>src;
       nodes++; //why nodes++ ? because given nodes value starts from 1....

       //this below code is for building adjacency matrix easy to solve....
       vector<vector<int>>adj(nodes+1);
```

```
 for (int i=0;i<edges;i++)
 {

        int p=mat[i][0];
        int q=mat[i][1];
        adj[p].push_back(q);
        adj[q].push_back(p);
 }
 //normal bfs on adjacency matrix....
queue<int>q;
int visited[1000000]={0};
vector<int>dist_ans(nodes,0);
q.push(src);
visited[src]=1;
int c=-1;
while(!q.empty())
{
    c++;
    int size=q.size();
    while(size--)
    {
        int x=q.front();
        q.pop();
        dist_ans[x]=c;
        for (int i=0;i<adj[x].size();i++)
        {
            if(visited[adj[x][i]])
            {
                continue;
            }
            else {
                visited[adj[x][i]]=1;
                q.push(adj[x][i]);
            }
        }

    }
}
     //for printing the output
for (int i=1;i<nodes;i++)
{
```

```
            if(i!=src)
            {
              if(dist_ans[i]==0)
              {
                  cout << "-1" <<" ";
              }
              else cout << (dist_ans[i]*6)<<" ";
            }
        }
        cout <<'\n';
    }
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Sample Test case 1

Input (stdin)

Download

```
2
4 2
1 2
1 3
1
3 1
2 3
2
```

Your Output (stdout)

```
6 6 -1
-1 6
```

EXPERIMENT-2.1(b)

1.Aim/Overview of the practical:

To demonstrate the concept of Tree Data Structures.

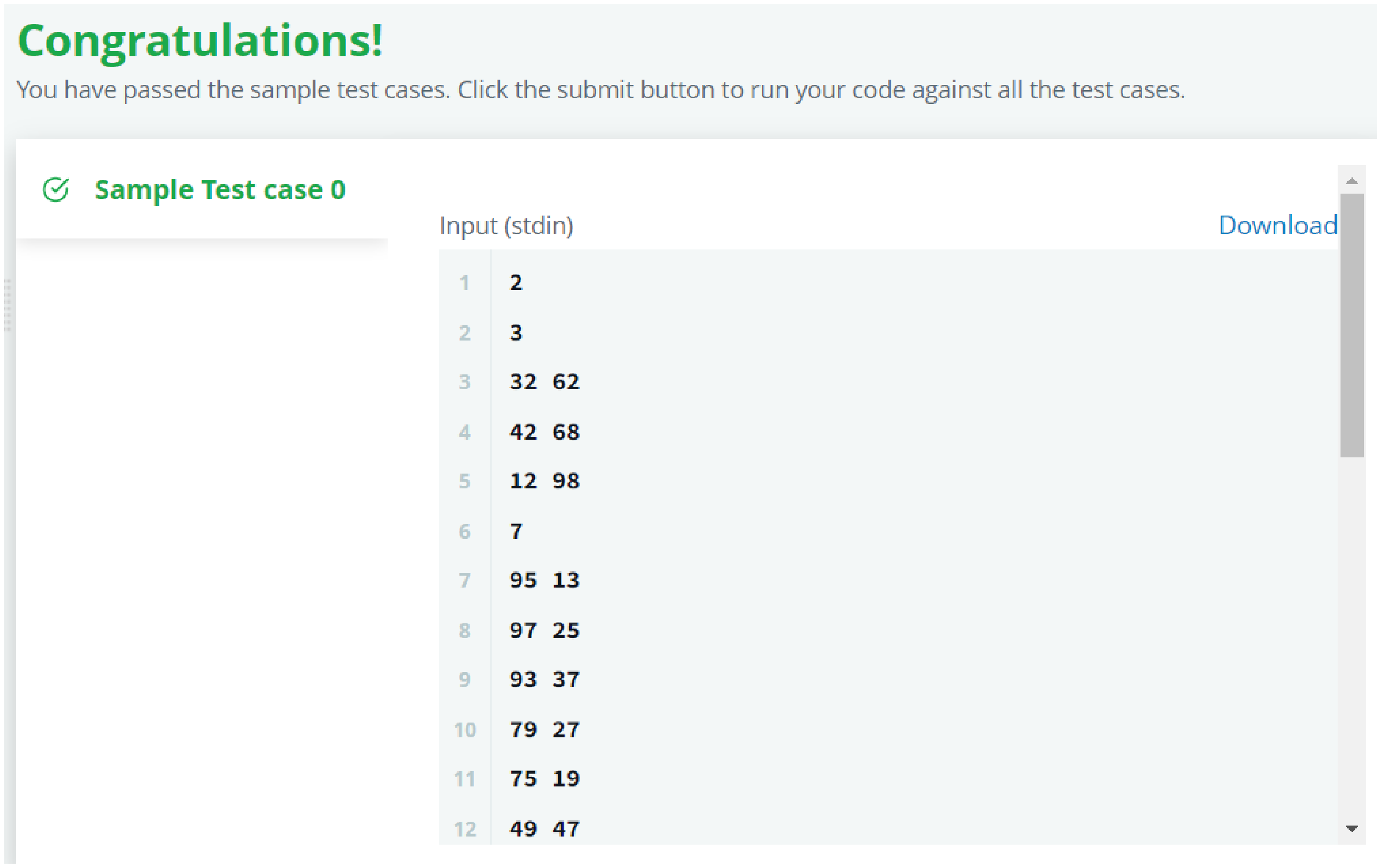
2. Task to be done/ Which logistics used:

https://www.hackerrank.com/challenges/the-quickest-way-up/problem?isFullScreen=true

3. Steps for experiment/practical/Code:

```
#include <bits/stdc++.h>
using namespace std;
int a, b, T, n, m;
int main()
{
    scanf("%d", &T);
    while (T--)
    {
        vector<pair<int, int> > graph[101]; // (n, d)
        int distance[101]={0}, snake[101]={0}, ladder[101]={0}, sp=0, lp=0;
        scanf("%d", &n);
        for (int i=0; i<n; i++)
        {
            scanf("%d%d", &a, &b);
            pair<int, int> p;
            p.first=b;
            p.second=0;
            graph[a].push_back(p);
            ladder[lp++]=a;
        }
        scanf("%d", &m);
        for (int j=0; j<m; j++)
        {
            scanf("%d%d", &a, &b);
            pair<int, int> p;
            p.first=b;
            p.second=0;
            graph[a].push_back(p);
            snake[sp++]=a;
        }
        sort(ladder, ladder+lp);
        sort(snake, snake+sp);
        lp=0;
        sp=0;
        for (int s=1; s<100; s++)
        {
            if (ladder[lp]==s)
                lp++;
            else if (snake[sp]==s)
                sp++;
            else {
                for (int i=1; i<=6; i++) {
                    if (s+i<=100)
                        graph[s].push_back(make_pair(s+i, 1));
                }
            }
        }
        for (int i=1; i<=100; i++)
            distance[i]=-1;
        pair<int, int> tpair;
        priority_queue<pair<int, int> > q;
        tpair.first=0;
        tpair.second=1;
        q.push(tpair);
        while (!q.empty())
        {
            tpair=q.top();
            q.pop();
            int min_dist=-tpair.first;
            int cur_node=tpair.second;
            if (distance[cur_node]==-1)
            {
                distance[cur_node]=min_dist;
                for (auto x : graph[cur_node])
                {
                    int next_node=x.first, dist=x.second;
                    if (distance[next_node]==-1)
                    {
                        pair<int, int> tp2;
                        tp2.first=-(min_dist+dist);
                        tp2.second=next_node;
                        q.push(tp2);
                    }
                }
            }
        }
        printf("%d\n", distance[100]);
    }
    return 0;
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin) Download

```
2
3
32 62
42 68
12 98
7
95 13
97 25
93 37
79 27
75 19
49 47
```

EXPERIMENT-2.1(c)

1.Aim/Overview of the practical:

To demonstrate the concept of Tree Data Structures.

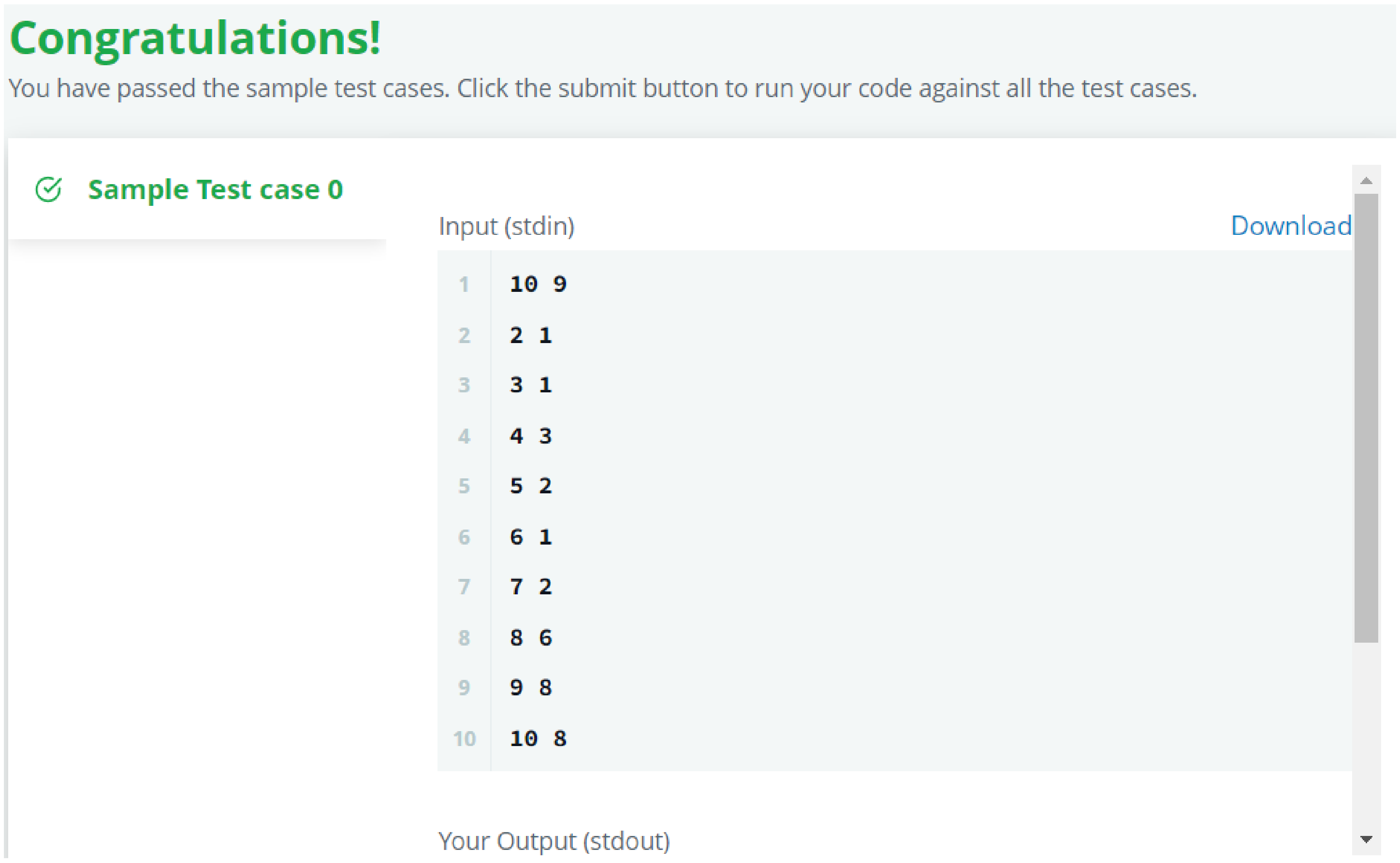
2. Task to be done/ Which logistics used:

https://www.hackerrank.com/challenges/even-tree/problem?isFullScreen=true

3. Steps for experiment/practical/Code:

```
#include<cstdio>
using namespace std;
int n,m,i,j,x,y,k,copii[104],tata[104],v[104],a[104][104];
int nod(int k,int t)
{
    for(int i=1;i<=a[k][0];i++)
        if(a[k][i]!=t)
            copii[k]+=1+nod(a[k][i],k);
    tata[k]=t;
    return copii[k];
}
int main()
{
    //freopen("input","r",stdin);
    //freopen("output","w",stdout);
    scanf("%d %d",&n,&m);
    for(i=1;i<=m;i++)
    {
        scanf("%d %d",&x,&y);
        a[x][0]++;
        a[y][0]++;
        a[x][a[x][0]]=y;
        a[y][a[y][0]]=x;
    }
    copii[1]=nod(1,0);
    for(i=1;i<=n;i++)
    {
        if(copii[i]%2==1&&tata[i]!=0)
        {
            tata[i]=0;
            k++;
        }
    }
    printf("%d",k);
    return 0;
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

```
10 9
2 1
3 1
4 3
5 2
6 1
7 2
8 6
9 8
10 8
```

EXPERIMENT-2.1(d)

1.Aim/Overview of the practical:

To demonstrate the concept of Tree Data Structures.

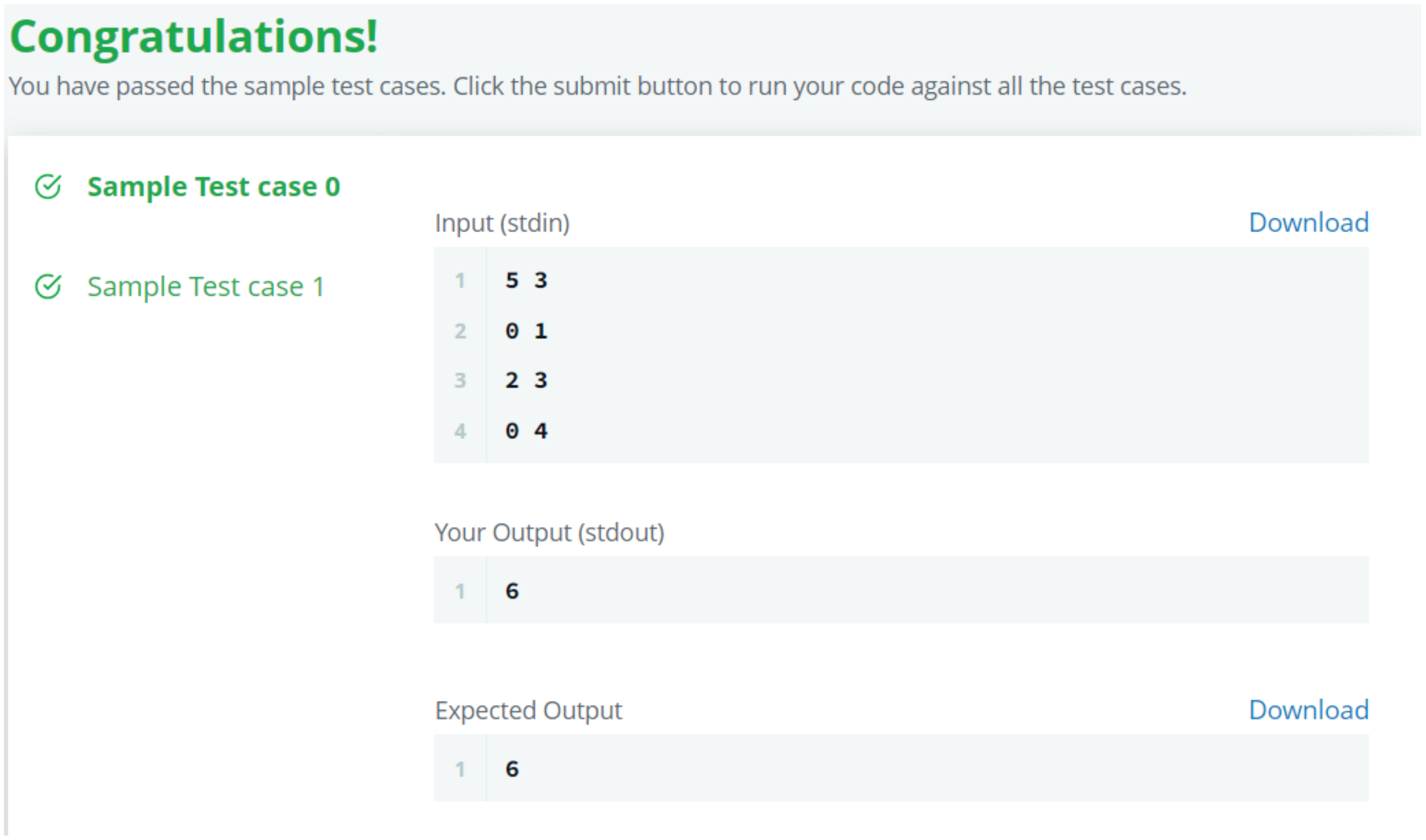
2. Task to be done/ Which logistics used:

https://www.hackerrank.com/challenges/journey-to-the-moon/problem?isFullScreen=true

3. Steps for experiment/practical/Code:

```
#include <vector>
#include <string>
#include <iostream>
#include <algorithm>

std::vector<unsigned int> pred;

int get_pred(int vertex) {

    if (pred.empty()) return 0;

    while(pred[vertex] != vertex) {
        vertex = pred[vertex];
    }
    return vertex;
}

int main() {
    int N, I;
    std::cin >> N >> I;

    // I hardcoded, killl meee
    if (N == 100000 && I == 2) {
        unsigned long long r = N;
        r *= (N - 1);
        r /= 2;
        r -= I;
        std::cout << r << std::endl;
        return 0;
    }

    /* MAKE DISJOINT SET ABSTRACTION*/
    pred.resize(N);
    for (unsigned int i = 0; i < N; i++)
        pred[i] = i;

    unsigned int a, b;
    for (int i = 0; i < I; i++) {
        std::cin >> a >> b;
        int ap = get_pred(a),
            bp = get_pred(b);

        if (ap < bp) {
            pred[bp] = ap;
        } else {
            pred[ap] = bp;
        }
    }

    /*
        Find the number of groups and the size of each group,
        but do it with a scope because why not?
    */
    std::vector<unsigned int> groups;
    {
        std::vector<unsigned int> freq(N, 0);
        for (int i = 0; i < N; i++) {
            freq[get_pred(i)]++;
        }

        for (auto& f : freq)
            if (f != 0) groups.push_back(f);
    }

    /*
        PREPARE THE RESULT
        It's summation from here on out, specialized summation.
    */
    unsigned long long result = 0;
    for (int i = 0, n = groups.size(); i < n - 1; i++)
        for (int j = i+1; j < n; j++)
            result += groups[i] * groups[j];

    std::cout << result << std::endl;

    return 0;
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Input (stdin)

```
5 3
0 1
2 3
0 4
```

Your Output (stdout)

```
6
```

Expected Output

```
6
```

EXPERIMENT-2.1(e)

1.Aim/Overview of the practical:

To demonstrate the concept of Tree Data Structures.

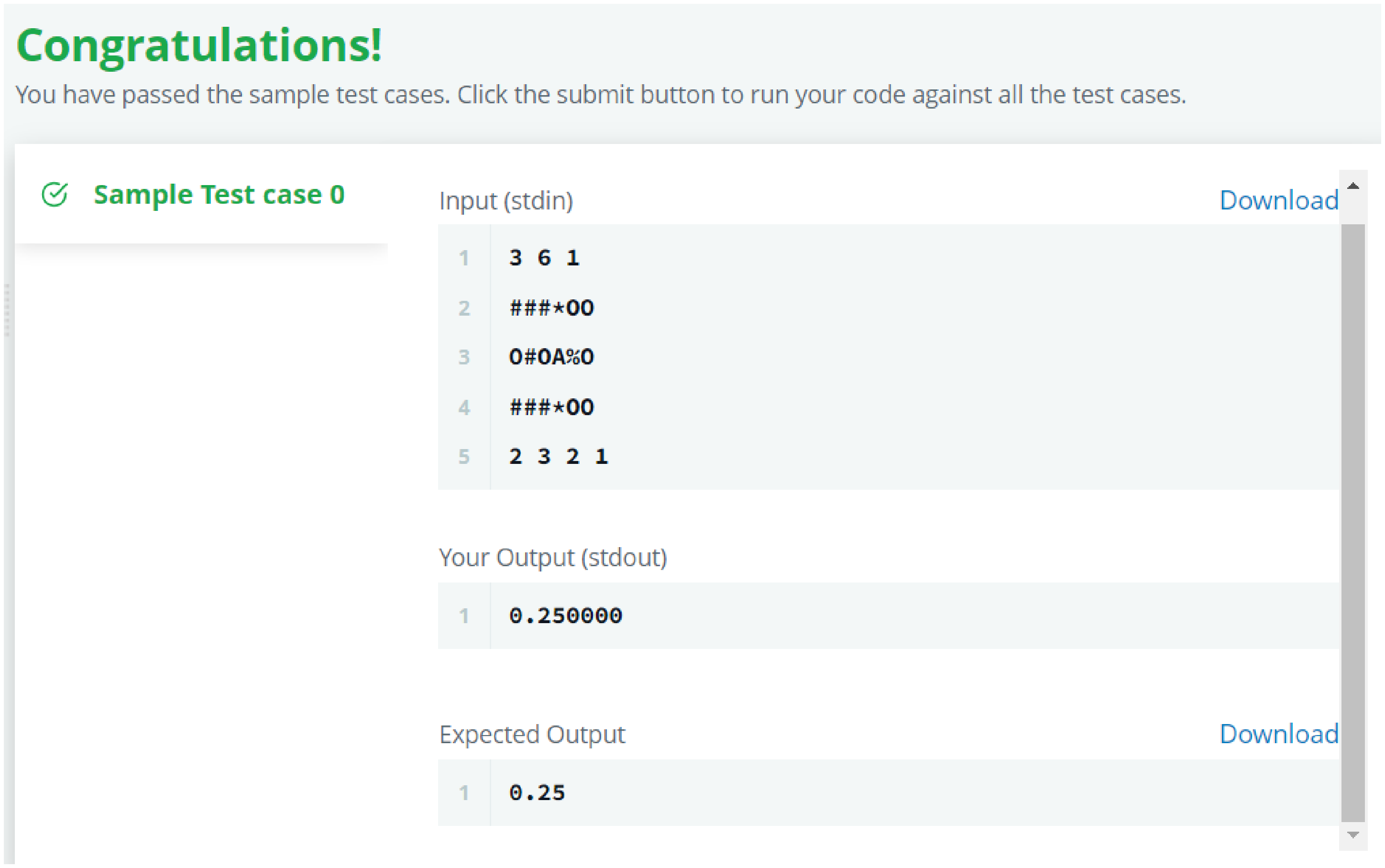
2. Task to be done/ Which logistics used:

https://www.hackerrank.com/challenges/frog-in-maze/problem?isFullScreen=true

3. Steps for experiment/practical/Code:

```
#include<cstdio>
char M[25][25]; // map
int T[25][25][2]; // tunnels
double P[2][25][25];
const int D[4][2] = {{-1,0}, {1, 0}, {0,-1}, {0,1}};
int h,w,t;
void calc(int in, int out) {
    for(int x=0;x<w;x++)
        for(int y=0;y<h;y++) {
            if(M[y][x] == '*' || M[y][x] == '#')
                P[out][y][x] = 0.0;
            if(M[y][x] == '%')
                P[out][y][x] = 1.0;
            if(M[y][x] == 'O' || M[y][x] == 'A') {
                int count = 0; double suma = 0.0;
                int px=x, py=y;
                if(T[y][x][0] != -1) {px = T[y][x][0]; py = T[y][x][1];}

                for(int i=0;i<4;i++) {
                    int x2 = px+D[i][0], y2 = py + D[i][1];
                    if(x2 < 0 || x2 >= w || y2 < 0 || y2 >= h)continue;
                    if(M[y2][x2] == '#')continue;
                    suma += P[in][y2][x2];
                    count++;
                }
                if(count == 0)
                    P[out][y][x] = 0.0;
                else P[out][y][x] = suma / count;
            }
        }
}

double get_ans(int p) {
    for(int i=0;i<h;i++)
        for(int j=0;j<w;j++)
            if(M[i][j] == 'A')
                return P[p%2][i][j];
    return -1.0;
}

int main() {
    scanf("%d%d%d", &h, &w, &t);

    for(int i=0;i<h;i++)
        scanf("%s", M[i]);

    for(int i=0;i<h;i++)
        for(int j=0;j<w;j++)
            T[i][j][0] = T[i][j][1] = -1;

    for(int i=0;i<t;i++){
        int x0, y0, x1, y1;
        scanf("%d%d%d%d", &y0, &x0, &y1, &x1);
        x0--;y0--;x1--;y1--;
        T[y0][x0][0] = x1;
        T[y0][x0][1] = y1;
        T[y1][x1][0] = x0;
        T[y1][x1][1] = y0;
    }

    const int limit = 80000;

    for(int i=0;i<limit;i++) {
        calc(i%2, (i+1)%2);
        // for(int y=0;y<h;y++){
        //     for(int x=0;x<w;x++)printf("%.3lf|", P[(i+1)%2][y][x]);
        //     printf("\n");
        // } printf("\n");
        //printf("%lf\n", get_ans(i+1));
    }
    printf("%lf\n", get_ans(limit));
}
```

4. Result/Output/Writing Summary:



Congratulations!

You have passed the sample test cases. Click the submit button to run your code against all the test cases.

Sample Test case 0

Input (stdin)

```
3 6 1
###*OO
O#OA%O
###*OO
2 3 2 1
```

Your Output (stdout)

```
0.250000
```

Expected Output

```
0.25
```

Learning outcomes (What I have learnt):

1. Through this experiment I learn concepts of Trees.

2. Different operations on Trees.

3.Learned about different algorithms of Tree data structures.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			