Homework #3

Due Time: 2017/12/14 (Thu.) 17:20
Contact TAs: `ada-ta@csie.ntu.edu.tw`

Instructions and Announcements

- There are **four programming problems** and **two handwritten problems**.
- **Programming.** The judge system is located at `https://ada-judge.csie.org`. Please login and submit your code for the programming problems (i.e., those containing "Programming" in the problem title) by the deadline. NO LATE SUBMISSION IS ALLOWED.
- **Handwriting.** For other problems (i.e., those containing "Handwriting" in the problem title), please submit your answers to the instructor at the beginning of the class. NO LATE SUBMISSION IS ALLOWED.
- **Collaboration policy.** Discussions with others are strongly encouraged. However, you should write down your solutions **in your own words**. In addition, for **each and every** problem you have to specify the references (e.g., the Internet URL you consulted with or the people you discussed with) on the first page of your solution to that problem. You may get zero point for problems due to the lack of references.
- Top-graded solutions/codes may be published as references for your fellow classmates.

Problem 1 - Flee (Programming) (10 points)

Problem Description

It's burning!

The house is burning! There are people still in the house. They need to flee from the building as soon as possible. There are emergency exits in the building, and each person can run to one of the exits to escape the fire.

Specifically, there are $R \times C$ rooms in the building arranged as a matrix consisting of R rows and C columns. We can denote the room at i-th row and j-th column as A_{ij}. For A_{ij} and $A_{i'j'}$, there's a door between them if and only if (($i = i'$ and $|j - j'| = 1$) or ($j = j'$ and $|i - i'| = 1$)). It takes exactly 1 second for a person, to run through a door. For example, from A_{11} to A_{12}, then to A_{22} takes exactly 2 seconds. (There's a door between A_{11} and A_{12}, so as A_{12} and A_{22})

However, some of the rooms have already in fire. That is, a person can not stay in or pass through these rooms.

Given the current situation of the building, you need to find out the minimum time to flee from the fire for each person.

Note that each room can contain an infinite number of people, and each door can be passed through by an infinite number of people at the same time.

Input

The first line contains two positive integers R, C indicating that the building contains $R \times C$ rooms arranged as R rows and C columns. Following R lines each contains a string A_i indicating the situation of i-th row of rooms. If A_{ij} is `'P'`, there's a person at the room A_{ij}. If A_{ij} is `'E'`, there's an emergency exit at the room A_{ij}. If A_{ij} is `'F'`, room A_{ij} is already in fire. If A_{ij} is `'.'`, room A_{ij} is empty and safe to stay and pass through.

- $1 \leq R, C \leq 3000$
- $|A_i| = C$
- $A_{ij} \in$ `"PEF."`
- at least one A_{ij} is `'P'`(There's at least one person in the building)
- at least one A_{ij} is `'E'`(There's at least one emergency exit in the building)
- at most 10^5 A_{ij} is `'P'`(There are at most 10^5 people in the building)

Subtask 1 (20 %)

- There's exactly one A_{ij} = `'P'` and exactly one $A\_\{ij\}$ = `'E'` (There's only one person in the building, and one emergency exit in the building)

Subtask 2 (30 %)

- There's exactly one A_{ij} = `'E'` (There's only one emergency exit in the building)

Subtask 3 (50 %)

- No other constraint

Output

If there are K people in the building, please output K lines.

Starting from the first row, then starting from the first column, for each $A\_\{ij\}$ = `'P'`, output an integer indicating the minimum time to run to an emergency exit if he/she is able to reach an emergency exit. Otherwise, output `"Died"` in one line (without quoting mark).

Sample Input 1

```
2 2
P.
FE
```

Sample Output 1

```
2
```

Sample Input 2

```
3 3
PPP
...
P.E
```

Sample Output 2

```
4
3
2
2
```

Sample Input 3

```
3 3
P..
FF.
PFE
```

Sample Output 3

```
4
Died
```

Sample Input 4

```
3 3
P.E
FFF
P.E
```

Sample Output 4

```
2
2
```

Problem 2 - Maximum Non-cut (Programming) (10 points)

Problem Description

In graph theory, a *cut* partitions the vertices of a graph into two disjoint subsets. A cut determines a *cut-set*, which is the set of edges that have one endpoint in each of the two subsets. In a weighted graph, the value or weight of a cut is defined by the sum of the weights in the cut-set.

After learning what the cut problem is, Eddy wants to study the maximum s-t non-cut problem. A s-t non-cut is removing some edges such that vertices s and t are still connected. The weight is defined by the sum of the weight of the removed edges.

Given an undirected graph. Find the weight of the maximum s-t non-cut.

Input

The first line contains four integers n, m, s, t. Each of the next M lines contains three integers u_i, v_i, w_i.

- $1 \leq n, m \leq 3 \times 10^5$
- $1 \leq u_i, v_i \leq n$
- $0 \leq w_i \leq 10^9$
- $s \neq t$

Output

Find the weight of the maximum s-t non-cut. Output `-1` if there's no s-t non-cut.

Sample Input 1

```
4 4 1 4
1 2 10
2 4 9
1 3 11
3 4 5
```

Sample Output 1

```
19
```

Sample Input 2

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

Sample Output 2

```
4
```

Sample Input 3

```
3 1 1 3
1 2 10
```

Sample Output 3

```
-1
```

Note

First Sample test case

- The removed edges are numbered 1 and 2.

Second Sample test case

- The removed edges are numbered 1, 2 and 3.

Problem 3 - 3 Roads (Programming) (20 points)

Problem Description

In the ADA country, there are N cities and M undirected roads. The i-th road connects cites u_i and v_i.

Eddy is an engineer in th ADA country. He was assigned a task to improve the traffic between cities.

Eddy will add new roads to the ADA country by repeating the following instruction:

- Choose x and y ($x \neq y$) such that city y can be reached by traversing exactly 3 roads from city x, and add a road connecting cities x and y. It is not allowed to add a road if there is already a road connecting cities x and y.

Please help Eddy find the maximum number of roads that can be added.

Input

The first line contains an integer T, indicating the number of test cases. The first line of each test case contains two integers N, M. Each of the next M lines contains two integers u_i, v_i.

- $1 \leq N \leq 10^5$
- $0 \leq M \leq 10^5$
- $1 \leq u_i, v_i \leq N$
- The graph has no self-loops or multiple roads.
- The size of each testdata is smaller than 5 MB.

Output

Find the maximum number of roads that can be added.

Sample Input

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

Sample Output

```
5
7
```

```
2
```

Note

First test case

- Add a road connecting city 1 and city 4 (`1 - 2 - 3 - 4`).
- Add a road connecting city 2 and city 5 (`2 - 3 - 4 - 5`).

Second test case

- Add a road connecting city 1 and city 5 (`1 - 3 - 4 - 5`).
- Add a road connecting city 2 and city 5 (`2 - 3 - 4 - 5`).
- Add a road connecting city 3 and city 5 (`3 - 1 - 2 - 5`).
- Add a road connecting city 1 and city 4 (`1 - 2 - 5 - 4`).
- Add a road connecting city 2 and city 5 (`2 - 1 - 3 - 4`).

Problem 4 - Metropolitan (Programming) (20 points)

Problem Description

HH is the king of the HH kingdom. There are N cities in the HH kingdom and M roads between some pair of them. Since HH kingdom is on a flat ground, no two roads intersect with each other.

HH wants to develop a group of cities into a metropolitan. If the chosen cities are far from each other, the development won't be easy. Thus, the chosen cities must have strong connectivity. HH thinks that a group of cities have strong connectivity if there exists a road between every pair of cities in the group.

HH wants to make the metropolitan as large as possible. That is, he wants to find a group of cities with strong connectivity and the number of cities in the group is maximized.

Input

The first line contains two integers N, M indicating the number of cities in the HH kingdom and the number of roads between them. Following M lines each contains two integers u_i, v_i indicating that there's a road between city numbered u_i and city numbered v_i.

- $1 \leq N \leq 2 \times 10^5$
- $0 \leq u_i, v_i < N$
- $u_i \neq v_i$
- there's no self-loop
- there's at most 1 road between each pair of cities
- the given graph is a simple planar graph (that is, no two roads intersect)

Output

Please output one line indicating the maximum possible size of the metropolitan.

Sample Input 1

```
5 6
0 1
1 2
2 0
2 3
3 4
4 2
```

Sample Output 1

```
3
```

Sample Input 2

```
10 6
0 1
0 2
0 3
1 2
1 3
2 3
```

Sample Output 2

```
4
```

Problem 5 - Compiler (Handwriting) (20 points)

In most programing language, the expression is not simply from left to right but a tree called AST (Abstract Syntax Tree). But while compiling, we always transfer it to a DAG (Directed Acyclic Graph) by merging those nodes they express the same expression since we do not want to compute something twice.
To simplify the problem we assume that the AST is a binary tree, and each node has a attribute id. We say two nodes are equal if there id, left child, and right child are equal.
You task is give a algorithm to convert the AST to DAG by merging some equal nodes to minimize the number of nodes.

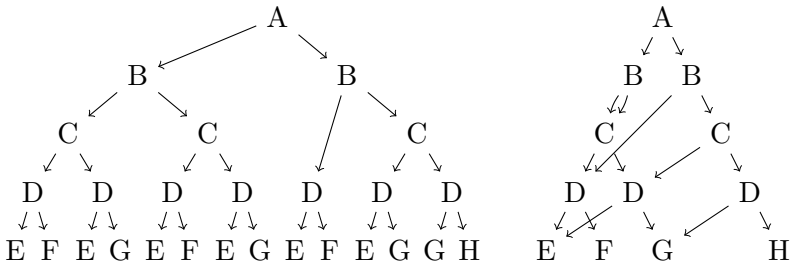
(1) (30%) Write pseudocode with the time complexity $O(N)$. Where N in number of nodes.

(2) (40%) Prove your algorithm can minimize the number of nodes.

(3) (30%) Prove time complexity.

Hint: Assume we have a perfect hash table. It is an amazing structure that can insert, search, and delete elements in time complexity $O(1)$. Note that the size of each element should be $O(1)$.

Sample:
AST: DAG:



Problem 6 - Minimum Spanning Tree (Handwriting) (20 points)

A long time ago, in a black forest, there was a country named Wololo Kingdom. Wololo Kingdom consisted of many villages. There were roads connected all villages up with each other. All of the roads were muddy and dangerous, so the king of Wololo Kingdom decided to pave some roads. He wanted to make it possible for people to move from one village to any other village without passing through a muddy road. Also, he wanted to spend as less gold as possible. This is equivalent to minimize the total length of roads to be paved.

The map of Wololo Kingdom is a connected graph $G = (V, E)$, where every vertex represents a village and every edge represents a road. Every edge is assigned a *positive* weight $w(u, v)$, specifying the length of the road that connects village u and v. Being a top engineer in Wololo Kingdom, you want to find a subset $T \subseteq E$ that connects all vertices and whose sum of edge weights is minimized.

(1) (15%) Please explain why T must form a tree and give the size of T. Your answer should contain an explanation of why T is acyclic.

(2) (15%) After you had found T and let the roads in T to be paved, you became famous for finding minimum spanning trees. The leader of the northern Wololo Kingdom also wanted you to find an MST for the northern region. The northern Wololo Kingdom can be represented by a subgraph $G' = (V', E') \subset G$, where $V' \subset V$ and $E' := \{E(u, v) | u, v \in V'\}$. However, you claimed there was no need of paving another road because all the paved roads within northern Wololo Kingdom $T' := \{T(u, v) | u, v \in V'\}$ were connected. Please substantiate your claim. That is, show that MSTs satisfy optimal substructure property. Specifically, prove that a subtree of MST is also optimal.

(3) (40%) A shepherd girl, Alice, came to you and claimed that she also successfully designed an algorithm to find T. Please prove it is correct, if her algorithm produces MST. Otherwise, give a counterexample.

```
ALICE'S ALGORITHM(G = (V, E), w)
1. T ← E
2. for each edge e ∈ E taken in non-increasing order of weight
3.     if T − {e} is a connected graph
4.         T ← T − {e}
5. return T
```

In question (4) and (5), the Prim's algorithm is implemented using Fibonacci heap whose time complexity is $O(|E| + |V| \log |V|)$.

(4) (15%) Since you were paying attention during class, you knew that you could use either Kruskal's algorithm or Prim's algorithm to find an MST. However, before receiving the map of Wololo Kingdom, you were worried about the time required to find an MST. To estimate the maximum time you would need, you assumed that the map of Wololo Kingdom is a complete graph. In terms of $|V|$, please compare the time complexity of algorithms of Kruskal and Prim running on a complete graph.

(5) (15%) Fortunately, you found that no roads cross each other on the map. The map is actually a planar graph, which is sparse. In terms of $|V|$, please compare the time complexity of algorithms of Kruskal and Prim running on a planar graph.

Hint: You can use the fact that the number of edges of a planar graph is at most $3|V| - 6$ when $|V| \geq 3$.