Homework #3

Due Time: 2016/12/15 (Thu.) 14:20
Contact TAs: ada01@csie.ntu.edu.tw

Instructions and Announcements

- There are two topic exercises and four regular problems. Topic exercises are easier than regular problems and are part of your mini-homeworks.
- **Programming.** The judge system is located at https://ada01-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.
- **Hand-written.** For other problems (also known as the "hand-written problems"), 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.

Topic Exercise 1: Graph Algorithms (Programming)

Time limit: **2**s

Description

You're given a weighted **undirected cyclic connected graph (UCCG)**. Please find a **minimum weight spanning undirected cyclic connected graph (MWSUCCG)** of it!

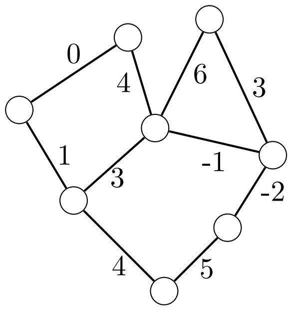
A **UCCG** is an undirected connected graph that contains **at least one cycle**.

A **sub-UCCG** is a subgraph that is also a **UCCG**.

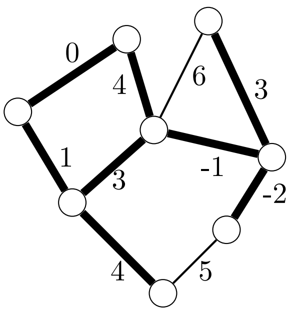
A **spanning sub-UCCG** is a **sub-UCCG** that contains all vertices of the original graph.

A **spanning sub-UCCG** is called a **MWSUCCG** if it has the minimum total (edge) weight among all of the **spanning sub-UCCG**s.

For example, consider the following UCCG.



Below is a MWSUCCG, with total weight 12.



Input Format

The first line contains two integers N, M, where N is the number of vertices, and M is the number of edges.

Each of the following M lines contains three integers a_i, b_i, w_i, indicating there are an (undirected) edge between vertex a_i and b_i, with weight w_i. Vertex numbers are 0-based.

Output Format

Please output one integer, the weight of the **MWSUCCG**.

Input Constraints

- $3 \le N \le 10^5$
- $3 \le M \le 5 \times 10^5$
- $0 \le a_i, b_i < N$
- $-10^9 \le w_i \le 10^9$
- There is no repeated edges and self-loops.
- The graph is connected, and contains at least one cycle.

- Test group 0 (0 points) : Sample testcases
- Test group 1 (11 points) : $N \le 10, M \le 20$
- Test group 2 (18 points) : $N \le 1000, M \le 2000$
- Test group 3 (7 points) : $N = M$
- Test group 4 (14 points) : All w_i are non-negative.
- Test group 5 (5 points) : All w_i are negative.
- Test group 6 (11 points) : There are exactly one edge with w_i negative.
- Test group 7 (34 points) : No additional constraints.

Sample Input 1

```
8 10
0 1 5
0 2 4
1 3 -2
2 4 3
3 5 1
4 6 4
4 7 6
3 7 3
5 6 0
3 4 -1
```

Sample Output 1

```
12
```

Sample Input 2

```
3 3
0 1 -1
1 2 -1
2 0 -1
```

Sample Output 1

```
-3
```

Sample Input 3

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

Sample Output 3

```
10
```

Problem 1

Flynn Rider is trying to solve a puzzle to rescue his beloved Rapunzel. In this puzzle he has an $n \times n$ board. Some of the squares are marked with obstacles, and one of which is a designated target square. The board also has two pawns on it. For each move, Flynn takes one of the pawns and moves it as far as it goes in one of the four cardinal directions until it bumps into an obstacle, the edge of the board, or the other pawn. His objective is to make some sequence of moves that ends with one of the pawns at the target square. For example, in the figure below, the moves shown allow him to bring pawn A to the target square in 6 moves.

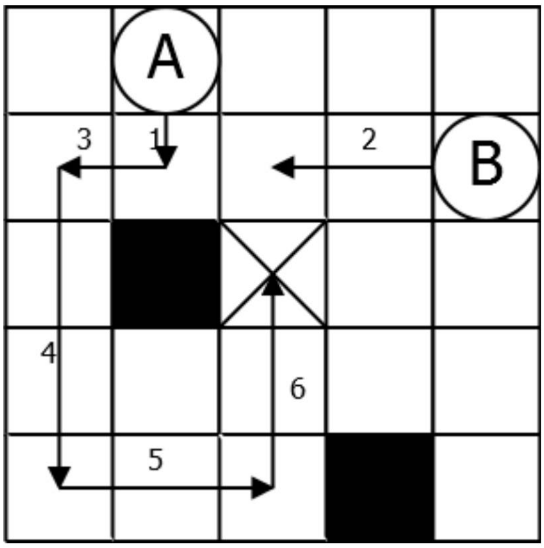


Figure 1: puzzle with two pawns

Given n, and the locations of the obstacles, pawns, and target, find an algorithm runs in time polynomial in n and determines whether or not it is possible for Flynn to solve this puzzle. What is the runtime of this algorithm?

(1) (6%) Consider the simplified version of the problem where there is only one pawn while all other conditions remain the same.

(2) (14%) Consider the original problem which has two pawns, both of which can be moved.

Hint: Relate this question to the reachability problem in some appropriate graph.

Problem 2

(1) (4%) According to Wikipedia:

> ... graph is said to be strongly connected if every vertex is reachable from every other vertex. The strongly connected components of an arbitrary directed graph form a partition into subgraphs that are themselves strongly connected ...

Please design an algorithm that decomposes a directed graph into its strongly connected components (SCC) by two depth first search (DFS). You can directly call DFS() as a subroutine.

(2) (10%) According to Wikipedia:

> ... 2-satisfiability (2SAT) is a computational problem of assigning values to variables, each of which has two possible values, in order to satisfy a system of constraints on pairs of variables ...

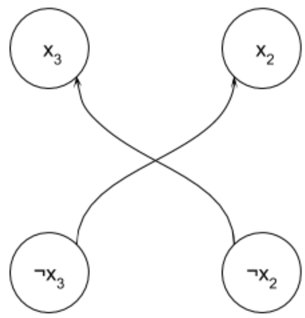
For example,

$$(\neg x_1 \vee \neg x_2) \wedge (x_3 \vee x_2)$$

is satisfiable since we can assign $x_1 = False, x_2 = True, x_3 = True$ to make the above expression evaluate to $True$. A 2SAT expression can be represented as a graph by transforming each clause to two implication forms and mapping every variable and negated variable to a vertex, every implication to an edge. For example,

$$(x_3 \vee x_2) \equiv (\neg x_3 \implies x_2) \equiv (\neg x_2 \implies x_3)$$

Then, $(x_3 \vee x_2)$ can be represented by



After converting a 2SAT problem to a graph, it can be solved by finding SCC. Please design an algorithm to solve a 2SAT problem by finding SCC and briefly justify the correctness of it. The input is a graph and you can directly call SCC() as a subroutine.

(3) (6%) Solve the following two 2SAT problems by drawing the implication graph and mark the SCC on it.

(a)

$$(\neg x_1 \vee \neg x_2) \wedge (x_3 \vee \neg x_2) \wedge (\neg x_1 \vee x_2) \wedge (x_3 \vee x_2)$$

(b)

$$(x_1 \vee x_3) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_3 \vee x_2) \wedge (\neg x_3 \vee \neg x_2)$$

Problem 3 - Brown South Forest - part 1 (Programming)

Time limit : 1s

Description

($30\% \cdot 50\% = 15\%$) There are two main attractions in the kingdom of Arendelle, namely, the Brown South Forest and the White North Mountain. In the Brown South Forest, there are many trails on which visitors could take a walk, enjoying the fresh air and the freedom in the natural surrounding.

There is a legend that these trials were built by an old wise man. One said that he hid some prophecy in the graph formed by these trails, but no one has ever figure out.

Leo, a college student who runs a special project abount Brown South Forest, is determined to solve this old mystery. After doing a field study, he observes that

- There are V sites and E trails in the Brown South Forest.
- Each trail starts from one of the sites and ends at a different site.
- All the trails are uni-directional, which is a rule since ancient times. There are signs to indicate the direction of each trail.
- For any two different sites u, v, There is at most one trail starting from u and ending at v.
- There are no loops. That is, there are no sequence of trails which form a path (see the definition below) that begins and ends at a same site.

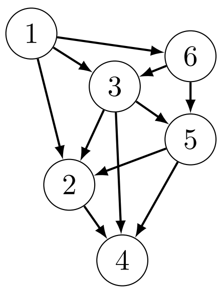
Leo suspects that some information is hidden in the number of different paths between two sites. A path is defined as a sequence of sites $v_1, v_2, \cdots, v_k$ such that there is a trail starting from v_i and ending at v_{i+1} for each i. Two paths $u_1, u_2, \cdots, u_k$, $v_1, v_2, \cdots, v_h$ is considered different if $k \neq h$ or there is an index i such that $u_i \neq v_i$.

Base on the map, Leo wants to calculate how many different paths that start from s and end at t for each posible s, t. Since the number could be large, he is only interested in the last digit of the number. So if there are 87 different paths, he would record as 7, and if there are no paths from a site to another, he would record as 0.

Not until Leo starts calculating does Leo realizes that it is not an easy job, since there are many sites and trails. After ten hours of hard works, he finally finishes calculating each pair of sites, and records the result on a paper.

But when he wakes up the next day, he couldn't find the paper on which he wrote the result. Clearly he doens't want to spend another ten hours again, so he asks if you have some clever idea. Can you help him to calculate based on his map?

For example, consider the following graph. The solution is the matrix $A = [a_{i,j}]$ in the right where $a_{i,j}$ is the last digit of the number of different paths from i to j. Notice that there are 11 different paths from 1 to 4, so $a_{1,4} = 1$.



$$\begin{bmatrix} 0 & 6 & 2 & 1 & 3 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 & 0 \\ 0 & 3 & 1 & 6 & 2 & 0 \end{bmatrix}$$

Input Format

The first line contains one integer V, which denotes the number of sites and the number of trails.
In the next V lines, each line contains V integers seperated by space. Let $e_{i,j}$ be the j-th number in the i-th line. $e_{i,j} = 1$ if there is a trail from i to j, or 0 otherwise.

- $1 \leq V \leq 300$

- Test group 0 (0 points) : Sample testcases.
- Test group 1 (10 points) : $V \leq 8$.
- Test group 2 (10 points) : $V \leq 50$ and the map forms a rooted tree.
- Test group 2 (10 points) : $V \leq 50$.
- Test group 3 (20 points) : No additional constraints.

Output Format

Output V lines, and each line contains V integers seperated by space. Let $a_{i,j}$ be the j-th number in the i-th line. $a_{i,j}$ should be the last digit of the number of different paths from i to j.

Sample Input 1

```
3
0 1 1
0 0 1
0 0 0
```

Sample Output 1

```
0 1 2
0 0 1
0 0 0
```

Sample Input 2

```
3
0 1 0
0 0 0
0 0 0
```

Sample Output 2

```
0 1 0
0 0 0
0 0 0
```

Sample Input 3

```
6
0 1 1 0 0 1
0 0 0 1 0 0
0 1 0 1 1 0
0 0 0 0 0 0
0 1 0 1 0 0
0 0 1 0 1 0
```

Sample Output 3

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

Problem 3 - Brown South Forest - part 2 (Programming)

Time limit : 1s

Description

(30% · 50% = 15%) You successfully helped Leo to calculate the results in the previous problem, and give him the map and the paper where the results is printed.

But in the next day, Leo turns to you for help again. After asking what happened, you learned that this time, Leo lost his map. The only thing left is the results you gave him yesterday. Leo wonders if it is possible to recover the original map based on the results you calculate. Could you help him this time?

Leo would accept any recovered map that matches the results. But the recovered map still has to satisfy the original constraints as the sites and trails in the Brown South Forest. The constraints are

- There are exactly V sites.
- Each trail should start from one of the sites and end at a different site.
- All the trails are uni-directional.
- For any two different sites u, v, there are at most one trail starting from u and ending at v.
- There are no loops. That is, there are no sequence of trails which form a path that begins and ends at a same site.

Input Format

The first line contains one integer V, which denotes the number of sites and the number of trails.
In the next V lines, each line contains V integers seperated by space. Let $a_{i,j}$ be the j-th number in the i-th line. $a_{i,j} = 1$ is the last digit of the number of different paths starting from i and ending at j.

- $1 \le V \le 300$

- Test group 0 (0 points) : Sample testcases
- Test group 1 (8 points) : $V \le 5$.
- Test group 2 (8 points) : $V \le 6$.
- Test group 3 (12 points) : $V \le 50$, and the original map formed a rooted tree.
- Test group 4 (14 points) : $V \le 50$.
- Test group 5 (8 points) : No additional constraints.

Output Format

Output V lines, and each line contains V integers seperated by space. Let $e_{i,j}$ be the j-th number in the i-th line. $e_{i,j}$ should be either 0 or 1, where 1 indicates that there is a trails start from i and end at j, and 0 indicates that there is no trails.
It is guaranteed that a solution exists. If there are multiple solutions, output any such.

Sample Input 1

```
3
0 1 2
0 0 1
0 0 0
```

Sample Output 1

```
0 1 1
0 0 1
0 0 0
```

Sample Input 2

```
3
0 1 0
0 0 0
0 0 0
```

Sample Output 2

```
0 1 0
0 0 0
0 0 0
```

Sample Input 3

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

Sample Output 3

```
0 1 1 0 0 1
0 0 0 1 0 0
0 1 0 1 1 0
0 0 0 0 0 0
0 1 0 1 0 0
0 0 1 0 1 0
```

Problem 4 - ADA kingdom (Programming)

Time limit : 1s

Description

(30%)

There are N cities in the ADA kingdom, numbered from 1 to N. City with number 1 is the capital of the kingdom.

There are two kinds of roads in the kingdom, one with length A and the other with length B. Every two different cities in the kingdom, including the capital, have a road between them, and the road is one of the two kinds.

Since the road with length A is special, there are just M of them in the kingdom.

Han Han is a traveler from a foreign country, and he arrived at the capital (city 1) yesterday. To plan a trip with the minimum travel distance, he wants to know the minimum distance from the capital to each city in this kingdom.

Can you calculate the answers for him?

Input Format

First line contains four integers N, M, A, B. For the following M lines, each line has two integers u_i, v_i which indicate that the road between city u_i and v_i is length A.

- $1 \leq N, M, A, B \leq 10^5$
- $1 \leq u_i, v_i \leq N$

- Test group 0 (0 points) : Sample test cases
- Test group 1 (15 points) : $1 \leq N \leq 300$
- Test group 2 (25 points) : $1 \leq N \leq 1000$
- Test group 3 (60 points) : $1 \leq N \leq 10^5$

Output Format

Please output N integers in one line, which are the minimum distance from capital to city i, i from 1 to N.

Sample Input 1

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

Sample Output 1

```
0 3 4
```

Sample Input 2

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

Sample Output 2

```
0 3 3 3 3 3
```

Sample Input 3

```
6 5 3 10
1 2
2 3
3 4
4 5
5 6
```

Sample Output 3

```
0 3 6 9 10 10
```

Sample Input 4

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

Sample Output 4

```
0 3 2 2
```

Sample Input 5

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

Sample Output 5

```
0 1 2 3
```