HW6

Rules & Instruction

- Compiler of programming problem.
  - C：`gcc -DONLINE_JUDGE -O2 -w -fmax-errors=3 -std=c11 main.c -lm -o main`
  - C++: `g++ -DONLINE_JUDGE -O2 -w -fmax-errors=3 -std=c++17 main.cpp -lm -o main`
  - Execution: `./main`
- Can I use theorems that haven't been mentioned in class?
  - Programming: No restriction. (無限制)
  - Handwritten: Please include its proof. (請寫上證明。)
- Can I refer to resources from the Internet or other sources that are not from textbooks or lecture slides?
  - Yes, but you must specify the references (the Internet URL you consulted with or the name and the page number of books). (可以，但必須附上參考來源，如網址或書名和頁數)
    - Handwritten: specify the references next to your solution. (手寫題請將參考附註於你的答案旁，你可以使用短網址工具)
    - Programming: specify the references at the top of your code in comments. (程式題請將參考以註解的方式附註於程式碼的最上方)
  - Although you can use external resources, it doesn't mean you can just paste the resources as your solution. **You have to answer by yourself**, and remember to specify the references; otherwise we'll view it as cheating. (雖然參考外部資源是允許的，但請用自己的話去作答，並切記要註記參考來源，否則會以作弊/抄襲處理)
- Rules of cheating.
  - Programming: We use tools to monitor code similarity. (我們用機器抓抄襲)
  - Handwritten: Though we forbid plagiarism, we still encourage you to discuss with each other. Remember that after discussion you must answer problems **in your own words**. (禁止抄襲，鼓勵大家有問題可以互相討論，**但請用自己的話去作答**。)
  - **Copycats will be scored 0.** You also need to have a cup of coffee with Professor Yeh. (**抄襲者與被抄襲者當次作業零分**，另外會跟教授喝杯咖啡☕ 。)
    - People who committed twice or more will be punished. (情節嚴重者，如累犯…，以校規處置)
- Rules of delay.
  NO LATE SUBMISSION IS ALLOWED.
  That is, zero toleration of late submission.

Non-Programming

Problem 1

Fill in the blanks $(30\ points)$
All the edges in the graph are stored with an adjacent matrix. Auxiliary space means the space except for storing graph.

For the complexity part, you should represent it using $|E|, |V|$.
$|E|$ means the number of **Edges**. $|V|$ implies the number of **Vertices**.

Shortest Path Algorithm	Design Method	Apply on graph containing negative edge	Apply on graph containing negative cycle	Time Complexity (tightest Big-O)	Auxiliary Space Complexity (tightest Big-O)
Bellman-Ford	(a) dynamic programming (b) greedy (c) divide and conquer	Yes / No	Yes / No		
Dijkstra (binary-heap)	(a) dynamic programming (b) greedy (c) divide and conquer	Yes / No	Yes / No		
Floyd-Warshall	(a) dynamic programming (b) greedy (c) divide and conquer	Yes / No	Yes / No		

Problem 2

BELLMAN-FORD(G, w, s)
1 INITIALIZE-SINGLE-SOURCE(G, s)
2 **for** $i = 1$ **to** $|G.V| - 1$
3 **for** each edge $(u, v) \in G.E$
4 RELAX(u, v, w)
5 **for** each edge $(u, v) \in G.E$
6 **if** $v.d > u.d + w(u, v)$
7 **return** FALSE
7 **return** TRUE

FLOYD-WARSHALL(W)
1 $n = W.rows$
2 $D^{(0)} = W$
3 **for** $k = 1$ **to** n
4 let $D^{(k)} = (d_{ij}^{(k)})$ be a new $n \times n$ matrix
5 **for** $i = 1$ **to** n

```
6            for j = 1 to n
7                d_ij^(k) = min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
8    return D^(n)
```

1. Briefly (within three sentences, or you will get no points) explain why the pseudo code of Bellman-Ford above can detect negative cycles? $(15\ points)$
2. Floyd-Warshall algorithm can also detect negative cycles on a directed graph, explain how to modify Floyd-Warshall to do this in $O(n^3)$ time complexity. $(15\ points)$
3. Modify the Floyd-Warshall algorithm to determine whether node v is reachable from node u for every pair u and v in $O(n^3)$ time complexity. $(5\ points)$

Problem 3

1. With a graph $G = (V, E)$ without multiple edges, prove or disprove $O(|V| + |E|) \in O(|V|^2)$. $(10\ points)$
2. Find a graph with negative edges and apply Bellman-Ford and Dijkstra algorithm on it. The answer graph must imply Dijkstra is not correct on a graph with negative edges. In other words, you should calculate the correct and wrong answer. (The number of vertices in your graph should be no greater than 5.) $(10\ points)$
3. Prove or disprove after applying the Dijkstra algorithm on a graph formed by distinct weight edges, and we will get a minimum spanning tree got by Prim's algorithm. And draw three example graphs, one is the original graph, another is the graph after applying Dijkstra, and the other is the minimum spanning tree got by Prim's algorithm. (The number of vertices in your graph should be no greater than 5. You also should specify the weight of every edge and the ID of every vertex.) $(15\ points)$

Programming

pA: Da-Hei-Hei Metro!

> Time Limit: 1s
> Memory Limit: 262144 KB

Description

Recently, Da-Hei-Hei is addicted to a game where players can freely construct the metro network. Since Da-Hei-Hei likes everything related to "cycle," his metro network is always full of **circular line**.

The costs are various between every circular line. Notably, some of them are too cheap so that their profit is severely low, and can't make him rich.

So now Da-Hei-Hei is searching for the cheapest circular line and wants to tear it down. Write a program to find out the most inexpensive circular line, and output its length.

P.S. The cost of a path on a metro network is proportional to the route's total distance.

Input Format

The first line contains two number N, M representing the number of the metro station, and the number of directional metro lines, respectively.

For next M lines, each of them contains three numbers u_i, v_i, w_i representing there is one metro line from station u_i to v_i with distance w_i.

For all test data, it is guaranteed:

- $0 < N \leq 300$
- $0 < M \leq N^2$
- $1 \leq u_i, v_i \leq 300 \ \forall i$
- $0 < w_i \leq 10^3$

Subtask1 (20%)

- $0 < N \leq 15$

Subtask2 (80%)

- No other restrictions.

Output Format

Output the minimum distance of the circular line you find, and if there is no circular line, please output "-1".

Sample Input

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

Sample Output

```
12
```

Sample Input

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

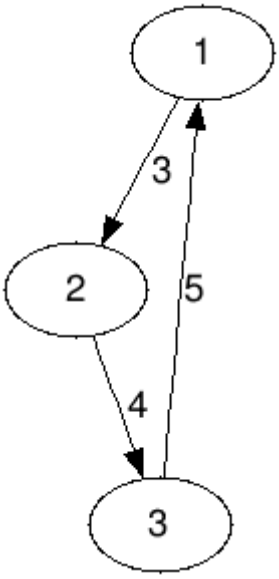
Sample Output

```
-1
```

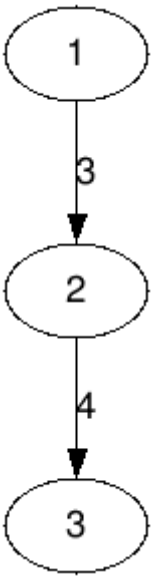
Hint

Handwritten Problem 2-2

For sample input 1



For sample input 2



pB: Traveling Salesman Problem

> Time Limit: 1s
> Memory Limit: 262144 KB

> It's okay for you to use data structure from WEB if you specify the reference.

Description

As a salesman, Grace likes riding a bike to visit her customers. Because of the long distance, she decides to add two stops(S_1, S_2) between the way of visiting the customer. Moreover, she also wants to calculate the distance back home. In other words, Grace wants to know the minimum distance from home(H) -> S_1 -> S_2 -> customer(C) -> home(H).

Input Format

The first line contains two number N, M representing the number of the stops and the number of bidirectional roads respectively.

The second line contains four numbers H, S_1, S_2, C, representing the ID of home, the first stop, the second stop, and the customer, respectively.

For next M lines, each of them contains three numbers u_i, v_i, w_i representing there is one road between stops u_i and v_i with distance w_i.

For all test data, it is guaranteed:

- $4 \leq N \leq 10^5$
- $0 < M \leq 3 \times 10^5$
- $1 \leq u_i, v_i, H, S_1, S_2, C \leq N \; \forall i$
- $H \neq S_1, H \neq S_2, H \neq C, S_1 \neq S_2, S_1 \neq C, S_2 \neq C$
- $0 < w_i \leq 10^6$
- You can reach any stops from any others.

Subtask1 (15%)

- $4 \leq N \leq 10$

Subtask2 (85%)

- No other restrictions.

Output Format

The minimum distance from home(H) -> S_1 -> S_2 -> customer(C) -> home(H).

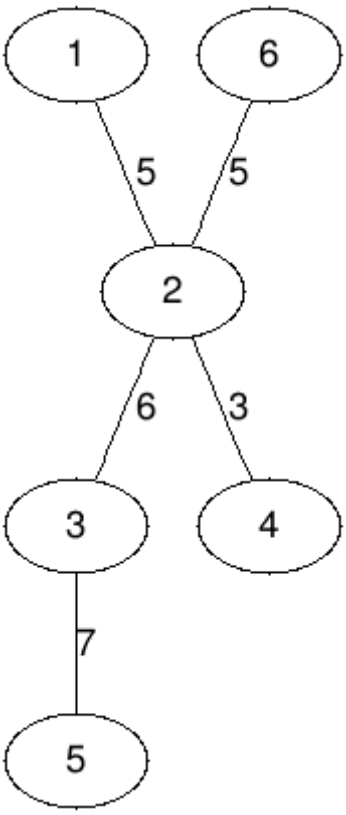
Sample Input

```
6 5
1 3 4 6
1 2 5
6 2 5
2 3 6
3 5 7
2 4 3
```

Sample Output

```
38
```

Hint



1 -> 3: 11
3 -> 4: 9
4 -> 6: 8
6-> 1: 10