Problem I
IOU

Problem ID: iou
CPU Time limit: 3 seconds
Memory limit: 1024 MB

Author: Godmar Back
Source: 2019 Virginia Tech Hi
School Programming Contest

Photo by Sharon McCutcheon on Unsplash

You are developing a new app intended to simplify expense-sharing among groups of friends. This app will allow them to keep track of who encountered an expense and how it should be shared with others through the form of IOUs. For instance, if Alice pays for a meal shared with Bob and Carol, and Bob's and Carol's shares were $5 and $10, respectively, then Bob would issue an IOU for $5 to Alice and Carol would issue an IOU for $10 to Alice.

Your app will maintain a ledger of who owes whom. Note that cycles can occur: For instance, if Bob initially owes Alice $10 and later pays a $5 expense on behalf of Alice, Alice would issue an IOU for $5 to Bob. This IOU would then cancel out, or reduce, the IOU Alice holds from Bob from $10 to $5. It's also possible for cycles to involve more than 2 people.

Your app will be given a list of IOUs issued and settle them as much as possible by considering all cycles and reducing each debt in a cycle by the minimum amount of debt occurring in the cycle. After all cycles are considered and canceled, your app should output who owes whom how much. If there are multiple ways in which cancelation can occur, you may choose any of them as long as there are no cycles left at the end. However, you may not introduce IOUs between friends that never gave an IOU to each other, e.g., if Alice owes Bob money, and Bob owes the same amount to Carol, you cannot remove Bob from the picture and declare that Alice now owes Carol.

Input

The input consists of a single test case. The first line contains two integers n and m ($1 \leq n \leq 100, 0 \leq m \leq 10\,000$), where n denotes the number of friends and m denotes the number of IOUs issued. Friends are numbered 0 to $n-1$. This is followed by m lines containing three integers a, b, c ($0 \leq a < n, 0 \leq b < n, a \neq b, 0 < c \leq 1\,000$) denoting an IOU given by friend a to friend b for c dollars. Any friend i holds at most one IOU from any friend j ($i \neq j$), but friend i may hold an IOU from friend j at the same time that friend j holds an IOU from i.

Output

First, output a single number p, denoting the number of IOUs left after canceling all cycles. Then, on the following p lines, output the IOUs that are left in the same form in which they appear in the input (e.g. using 3 integers a, b, c denoting that friend a owes friend b c dollars). Do not include any IOUs fully canceled, i.e., all the IOUs you output must have $c > 0$.

Sample Input 1

```
4 5
0 1 10
1 2 10
0 3 10
3 2 10
2 0 20
```

Sample Output 1

```
0
```

Sample Input 2

```
2 2
0 1 20
1 0 5
```

Sample Output 2

```
1
0 1 15
```

Sample Input 3

```
4 5
0 1 10
1 2 10
0 3 10
3 2 10
2 0 10
```

Sample Output 3

```
2
3 2 10
0 3 10
```