**Algorithms Advanced with C#: Exam**

1. **Reaper Man**

Using Reapers is mandatory however this time you are not the one sending them to scout, you are the developer implementing the path finding logic behind the movement of this unit.  
You need to find the shortest path that the Reaper will use, keep in mind that there are obstacles along the way and units cannot move through those.

## Input

* The **first line** holds an integer **n** – the number of blocked paths, once a path is blocked other paths will start from there
* On the **second line**, you will receive the number **m** – the number of paths
* On the next line the **start** and **end** positions separated by space
* At the next **m** **lines**, you will receive the paths in the format: **{from} {to} {distance}**

## Output

* First line of output prints the path
* On the second line print the **total distance**

## Constraints

* Number **n** will be an integer in the range [**0**…**10000**]
* Number **m** will be an integer in the range [**0…10000**]
* The distances will be integers in the range [**0…10000**]
* All **n** will be numbered from **0** to **N - 1**.

## Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  6  0 3  0 1 1  0 1 2  1 2 3  1 2 1  2 3 5  2 3 2 | 0 1 2 3  5 |
| 5  5  1 3  0 1 1  1 2 3  2 4 5  2 3 3  1 4 14 | 1 2 3  6 |

“Five exclamation marks, the sure sign of an insane mind.”

― Terry Pratchett, Reaper Man