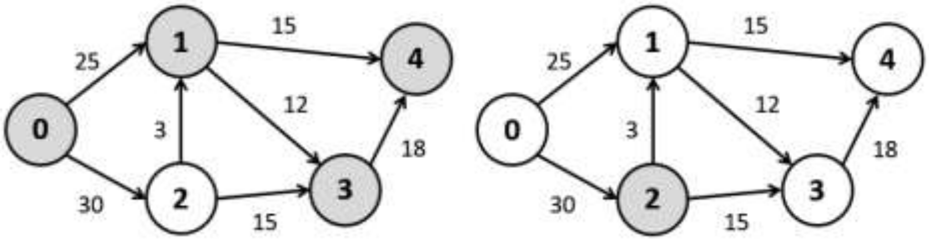
Problem 4 – Robbery

You are robber who just stole a TV. Now you must escape the cops without being caught. You are given a map of the city streets. However, the TV is quite heavy and you have **limited energy E**. There a few rules:

- Going from one point to another costs you some **energy** (displayed as a value on each arrow) and takes one turn.
- Each point is being watched by a **video camera**. A point can be **black** (a camera is **not** watching it) or **white** (a camera is watching it). Every turn points change color to the opposite (from black to white and vice versa). See the example below.
- You can only travel to points where the **camera is currently on** (because when you step on it the camera will be off and you will not be caught).
    - After stepping on a point you can **wait 0 or more turns** before going further (the camera will not catch you). Note that every turn you wait costs **W energy**.

Find the path that requires the **least energy** to go to the final point. Print the required energy.



At each turn points change color as shown above

Input

- On the first line you will receive all points in the format "**<node1><color1> <node2><color2> …<nodeK><colorK>**".
- On the second line, you will receive the **starting energy**.
- On the third line, you will receive the cost (in energy) for **waiting one turn**.
- On the fourth line, you will receive the **starting node**.
- On the fifth line, you will receive the **ending node**.
- On the sixth line you will receive a number - **n** specifying the **number of point connections**.
- On the next **n** lines you will receive the connections in the format "**<start> <end> <distance>**"

Output

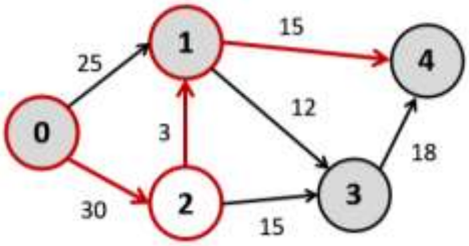
- Print the **amount of energy you have** at the end.
- If you do not have enough energy, print how much energy you still need to reach the end in the format "**Busted - need *{x}* more energy**".

Constraints

- The amount of points in the city will be between **[2…20000]**.
- The starting energy **E** will be a valid integer between **[0…100000].**
- The waiting cost **W** will be a valid integer between **[0…10000].**
- The distance of a connection will be a valid integer between **[0…10000].**
- The points will always be numbers starting from **0**.
- The color will be either "**b**" or "**w**" – **b** means the camera is currently **not** watching, **w** means the camera is currently watching.
- There will always be a valid path from **start** to **end**.
- Time limit: **100 ms**. Allowed memory: **32 MB**.

Examples

Input	Output	Visual

`0b 1b 2w 3b 4b`
`70`
`10`
`0`
`4`
`7`
`0 1 25`
`0 2 30`
`1 3 12`
`1 4 15`
`2 1 3`
`2 3 15`
`3 4 18`
	`12`	 0
		1
		2
		3
		4
		25
		15
		12
		3
		18
		30
		15
		Comments
		Start from 0.
		• Going to 2 costs 30 energy.
		• Going to 1 costs 10 (for waiting one turn) + 25 = 35 energy
		From 2 we go directly to 1 with energy 3.
		From 1 we need to wait 1 turn for both 3 and 4 to become white and then go for 4 (10 + 15 = 25 energy).

Input	Output

`0b 1w 2w 3w 4w 5b 6b 7w 8b 9w 10b 11b 12w 13w`
`99`
`5`
`1`
`11`
`18`
`1 0 5`
`1 2 17`
`1 3 22`
`3 4 17`
	`Busted - need 75 more energy`
	Comments
	*(Path is **1 -> 3 -> 5 -> 6 -> 8 -> 9 -> 10 -> 12 -> 13 -> 11**)*

3 5 4
5 6 3
5 7 12
6 4 7
6 8 31
7 2 5
7 10 117
8 9 44
9 10 2
10 6 9
10 12 29
11 9 1
12 13 16
13 11 8	