Rocket Monkey

A monkey is making a living in a forest with N trees. The trees are numbered from 1 to N. Monkey wants to move from tree 1 to tree N. There are tigers on the ground and the monkey has to jump from tree to tree to get there. You are given M pairs of trees for which the monkey can jump directly in either direction and you are also given the distance between two trees in pairs. If the monkey jumps from a tree to another tree (the two trees are in the M pairs of course) by a distance of x, then the elevation (the vertical height of the monkey's location) decreases by x. That is, if the monkey is at elevation h of the first tree and jumps to another tree by a distance of x, then the monkey will be at elevation h-x when it arrives at the second tree. So, the height of the first tree H has to be large enough so that H-x is at least 1, for the jump to be possible. (The fact that a pair of trees are in the given M pairs does not guarantee that this height condition is always satisfied.) H-x should be at least 1 because being at the elevation of 0 is the same as being on the ground. In other words tigers can reach elevation below 1. Take care that the monkey has to jump from an elevation at least x+1 to reach the second tree without touching the ground. The monkey can crawl on a tree upwards or downwards.

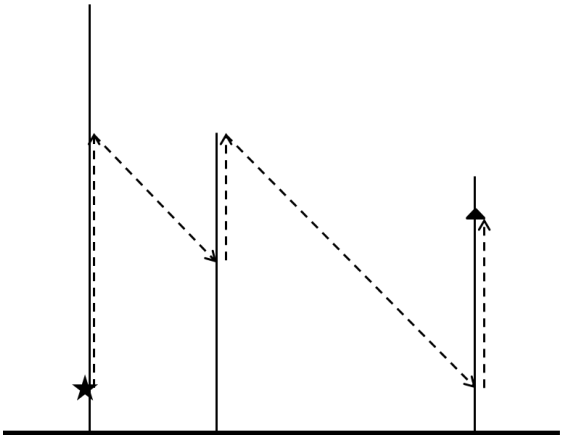
It takes time x to jump from a tree to another tree by a distance of x. It takes time y to crawl up or down a distance of y.

Finally, the monkey has one rocket which it can use in at most one jump. If it uses the rocket the elevation does not go down but stays the same during the jump. (The time for the jump is the same.) Even if the monkey uses the rocket, the jump can be made only between the trees that are in one of the given M pairs. Notice that the rocket allows the monkey to make a jump that is disallowed by the height condition mentioned in the first paragraph.

Note that the monkey may not use the rocket at all and the answer (for this problem) may be one of such cases.

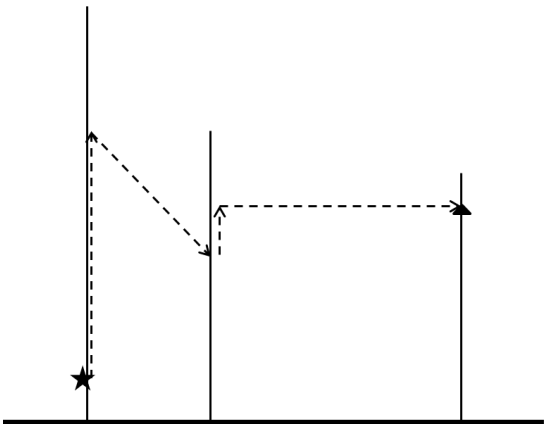
The monkey is initially at elevation s in tree 1 and wants to move to elevation t at tree N.

See the example in the following figure. There are 3 trees, numbered 1, 2, and 3 from the left. There are 2 pairs of trees (1, 2) and (2, 3) that the monkey can jump between and the distances are 3 and 6, respectively. The heights of the trees are 10, 7, and 6, respectively in the numbered order. The monkey's initial elevation is 1 and marked with a star. The target elevation is 5 and is marked with a triangle.

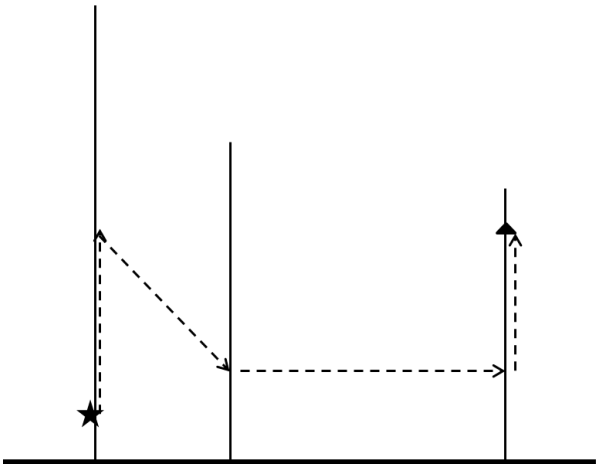


The arrows indicate a method of moving to the target without using the rocket. The time would

be 6+ 3+ 3+ 6+ 4=22. This is one of the ways with the minimum time without using the rocket.



The figure above shows a method that uses the rocket once. The time would be 6+ 3+ 1+ 6=16. This is one of the ways with the minimum possible time.



The figure above shows a slightly different method with same total time.

Given the heights of the trees and the pairs of trees that the monkey may jump between, write a program to compute the minimum possible time to move from elevation s of tree 1 to elevation t of tree N.

[Input]
The first line of the input file contains a number T (T ≤150) which indicates the number of test cases. The first line of each test case contains two integers N and M, which represent the number of trees and the number of tree pairs ($2 \le N \le 100$, $1 \le M \le 500$). The next line contains the tree heights ordered by tree numbers. The heights are integers between 1 and 100,000, inclusive. Each of the next M lines contains one of the M pairs by two tree numbers a and b and the distance x between trees a and b. A same pair will be given just once. The last line contains s and t in that order.

There are 3 kinds of inputs as follows.

- Set 1: $2 \le N \le 10$, $1 \le M \le 20$, the tree heights are between 1 and 100, inclusive.
- Set 2: A method that does not use the rocket will be one of the answers.

- Set 3: No further restrictions other than the original ones.

[Output]
For each case, your program has to output the minimum possible time to move from the initial position to the target position. In case it is not possible to move to the target position, your program should print a -1.

[Examples]
Input

```
3
3 2
10 7 6
1 2 3
2 3 6
1 5
2 1
5 5
1 2 3
5 4
2 1
5 5
1 2 3
5 3
```

Output

```
16
4
4
```