Permutation Puzzle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	512 megabytes

Little relyt871 is solving a puzzle. The key to the puzzle is a permutation containing numbers $1 \dots n$. The values at some positions of the permutation are already fixed, and relyt871 needs to fill numbers into the remaining positions.

Besides, little relyt871 has gathered m extra requirements about the permutation. Let the solution be represented as $p_1, p_2, ..., p_n$, each clue is a pair of indices (u_i, v_i), which means that $p_{u_i} < p_{v_i}$ should be satisfied in the solution.

Little relyt871 wonders if all requirements may be satisfied at the same time. Write a program to find a valid solution when there is one.

Input

The first line of the input contains the number of test cases T ($1 \le T \le 20\,000$).

For each test case:

- The first line contains two integers n, m ($2 \le n \le 200\,000, 1 \le m \le 500\,000$).
- The second line contains n integers $p_1, p_2, ..., p_n$ ($0 \le p_i \le n$). If $1 \le p_i \le n$, then the value at position i is fixed as p_i, otherwise it is your task to determine the value at position i. It is guaranteed that for $1 \le i < j \le n$, if $p_i > 0$ and $p_j > 0$, then $p_i \ne p_j$.
- The following m lines each contains two integers u_i, v_i ($1 \le u_i, v_i \le n$), denoting the clues. It is guaranteed that the clues don't contradict themselves. Formally, there doesn't exist a list of clues $(u_{i_1}, v_{i_1}), (u_{i_2}, v_{i_2}), ..., (u_{i_k}, v_{i_k})$ such that $v_{i_j} = u_{i_{j+1}}, 1 \le j < k$ and $v_{i_k} = u_{i_1}$.

The sum of n over all test cases doesn't exceed 200 000, and the sum of m doesn't exceed 500 000.

Output

For each test case:

- If there exists no valid solution, output "`-1`" in a single line.
- Otherwise, output one line containing n integers seperated by spaces, denoting the solution. If there are multiple solutions, print any.

Examples

standard input	standard output
2 4 4 1 0 0 4 1 2 1 3 2 4 3 4 3 2 0 3 1 1 2 3 1	1 3 2 4 2 3 1
1 4 4 1 4 0 0 1 2 1 3 2 4 3 4	-1