Problem D. Filesystem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The Red Monster is preparing a contest for the Overly Complicated Problem Colloquium. The Red Monster has a number of files that need to be uploaded to the contest preparation system, Polytope. All of the files are in the same folder; each file has a creation date and a file name. All file names and creation dates are distinct.

The interface for uploading files looks like this:

File name	Creation date
checker.cpp	17.03.2023
clever_generator.cpp	18.09.2022
doit.sh	15.12.2022
examples.txt	10.12.2021
generator.cpp	07.05.2021
monsters.Inc.3D.2001.1080p.BluRay.Half-OU.DTS-ES.x264-HDMaNiAcS.avi	17.10.2021
not_a_virus.exe	07.06.2022
sol.cpp	18.06.2021
spxG7HoMSMHH225xjadbnA.tmp	06.07.2023
tutorial.tex	03.02.2021
xxx_my_password_DO_NOT_HACK.docx	10.01.2023
validator.cpp	15.01.2023

In one operation, the Red Monster does all of the following:

- Sorts the files in the folder, either alphabetically by file name or by the creation date.
- Chooses a contiguous segment of files and uploads those files to Polytope.

Note that files are not deleted after they are uploaded. Only a subset of files has to be uploaded to Polytope. The other files may be embarrassing and therefore must not be uploaded. For example, in the table above, only the files with bold file names should be uploaded; the rest have clearly nothing to do with the contest.

Each file must only be uploaded once, that is, there should not be any file that is uploaded in several operations. What is the minimum number of operations needed to upload exactly the necessary files?

Input

The first line contains one integer t ($1 \le t \le 1000$) — the number of test cases. t test cases follow. Each test case is described as follows.

Let n be the total number of files. We index the files $1 \dots n$ and assume that the order of files when sorting by file name is $1, 2, 3, \dots, n$.

The first line of the the test case consists of two integers n and k ($1 \le k \le n \le 1000$) — the total number of files and the number of files that need to be uploaded.

The second line of the the test case consists of k integers $u_1, u_2, \dots, u_k$ ($1 \le u_i \le n$, for all i; all u_i are pairwise distinct). These are the indices of the files that must be uploaded.

The third line consists of a permutation $p_1, p_2, \dots, p_n$ of $1 \dots n$. This indicates that the order of files when sorted by creation date is $p_1, p_2, \dots, p_n$.

It is guaranteed that the sum of n over all test cases doesn't exceed 1000.

Output

For each test case, print the answer on a separate line — a single integer, the minimum number of operations needed to upload all files.

Example

standard input	standard output
2 12 8 2 5 8 3 4 10 12 1 10 5 8 6 4 7 2 3 11 12 1 9 8 4 1 3 5 7 1 4 5 8 7 6 3 2	3 4

Note

The first example test case corresponds to the example in the statement. Ordered by creation date, it looks as follows:

ID	File name	Creation date
10	**tutorial.tex**	03.02.2021
5	**generator.cpp**	07.05.2021
8	**sol.cpp**	18.06.2021
6	monsters.Inc.3D.2001.1080p.BluRay.Half-OU.DTS-ES.x264-HDMaNiAcS.avi	17.10.2021
4	**examples.txt**	10.12.2021
7	not_a_virus.exe	07.06.2022
2	**clever_generator.cpp**	18.09.2022
3	**doit.sh**	15.12.2022
11	xxx_my_password_DO_NOT_HACK.docx	10.01.2023
12	**validator.cpp**	15.01.2023
1	**checker.cpp**	17.03.2023
9	spxG7HoMSMHH225xjadbnA.tmp	06.07.2023

Observe that if we only ever sorted by the file name, we would need to use 4 operations. The same holds if we only ever sorted by creation date. A solution in 3 operations looks as follows:

- Sort by file name. Upload files 2, 3 and 4 (clever_generator.cpp, doit.sh and examples.txt). Note that if we also uploaded file 5 (generator.cpp) in this step, we would mess up the next step.
- Sort by creation date. Upload files 10, 5 and 8 (tutorial.tex, generator.cpp and sol.cpp).
- Sort by creation date. Upload files 12 and 1 (validator.cpp and checker.cpp).

There are other solutions with 3 operations. It can be proven that there is no solution with 2 operations.

In the second example test case, we want to upload exactly the files with odd indices. Whichever way we sort, there is never even a situation where two files we want to upload are consecutive. Therefore, every file must be uploaded separately.