D. Inaccurate Subsequence Search

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Maxim has an array a of n integers and an array b of m integers ($m \le n$).

Maxim considers an array c of length m to be good if the elements of array c can be rearranged in such a way that at least k of them match the elements of array b.

For example, if $b = [1, 2, 3, 4]$ and $k = 3$, then the arrays $[4, 1, 2, 3]$ and $[2, 3, 4, 5]$ are good (they can be reordered as follows: $[1, 2, 3, 4]$ and $[5, 2, 3, 4]$), while the arrays $[3, 4, 5, 6]$ and $[3, 4, 3, 4]$ are not good.

Maxim wants to choose every subsegment of array a of length m as the elements of array c. Help Maxim count how many selected arrays will be good.

In other words, find the number of positions $1 \le l \le n - m + 1$ such that the elements $a_l, a_{l+1}, \ldots, a_{l+m-1}$ form a good array.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains three integers n, m, and k ($1 \le k \le m \le n \le 2 \cdot 10^5$) — the number of elements in arrays a and b, the required number of matching elements.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$) — the elements of array a. Elements of the array a are not necessarily unique.

The third line of each test case contains m integers $b_1, b_2, \ldots, b_m$ ($1 \le b_i \le 10^6$) — the elements of array b. Elements of the array b are not necessarily unique.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$. Similarly, it is guaranteed that the sum of m over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output the number of good subsegments of array a on a separate line.

Example

input

```
5
7 4 2
4 1 2 3 4 5 6
1 2 3 4
7 4 3
4 1 2 3 4 5 6
1 2 3 4
7 4 4
4 1 2 3 4 5 6
1 2 3 4
11 5 3
9 9 2 2 10 9 7 6 3 6 3
6 9 7 8 10
4 1 1
4 1 5 6
6
```

output

```
4
3
2
```

```
4
1
```

Note

In the first example, all subsegments are good.

In the second example, good subsegments start at positions 1, 2, and 3.

In the third example, good subsegments start at positions 1 and 2.
