You are given a string, **S** (containing only lower-case letters). Next you are given some queries. The queries are of the form:

- **K M**

This means that you need to find the **M**th (1-based) substring from the list of sorted distinct substrings of **S** which has length exactly equal to **K**. For example, say **S** = "abdcabdc" and we are processing the query **K = 4**, **M = 2**, that means we are looking for substrings of length **4**. They are:

1. abdc
2. bdca
3. dcab
4. cabd
5. abdc

Since we are looking for distinct substrings, the second "abdc" will be ignored. Now if we sort them the substrings will look like:

1. abdc
2. bdca
3. cabd
4. dcab

So for **M = 2**, the output would be "bdca". However for **K = 4** and **M = 4**, the output would be "dcab". But you don't need to output the actual string. Rather just output the starting index (0-based) of the output string. If there are multiple possible answer, then output the lowest one. So for **K = 4 and M = 1** (output string "abdc"), you can see that it can be found in two different starting indices, 0 and 4. As 0 is lowest, so you need to output 0.

## Input

First line will contain one integer, **T (T ≤ 10)**, number of test cases. Each case starts with a line containing **S (1 ≤ |S| ≤ 100000)**. Next line will contain **Q (1 ≤ Q ≤ 100000)**, number of queries. Each query will contain two integers **K** and **M (1 ≤ K ≤ |S|, 1 ≤ M ≤ 100000)** in a line.

## Output

For each query, output the starting index (0-based) of the desired substring. If there is no answer, then output "Not found". See sample for clarification.

## Sample Input

| Sample Input | Output for Sample Input |
|---|---|
| 1<br>abdcabdc<br>13 | 0<br>1<br>3 |

| | |
|---|---|
| 1 1 | 2 |
| 1 2 | Not found |
| 1 3 | 0 |
| 1 4 | 1 |
| 1 5 | 3 |
| 2 1 | 2 |
| 2 2 | Not found |
| 2 3 | 1 |
| 2 4 | 2 |
| 2 5 | Not found |
| 4 2 | |
| 4 4 | |
| 4 5 | |