CS3230: Programming Assignment 1

Due: Sunday, 6th March 2022, 11:59:59 pm SGT.

This programming assignment is an **optional** component of the module, worth up to 2 bonus marks. Detailed submission instructions can be found at the last page.

Templates are provided for each language. These templates will provide a starting point for your implementation and also provide an implementation of fast input routines. You are recommended to use the templates provided to you. You have to submit your own work, and any form of plagiarism is **not tolerated** and will be subject to disciplinary action.

Background

You have been introduced to string pattern matching in Lecture 6, but sometimes the text and pattern strings may contain wildcard characters to allow errors in string matching. Here, we will focus on the `?` wildcard which represents any single character from the alphabet. On the other hand, the number of errors in the matching should be bounded, hence the number of wildcards used in the match cannot exceed some constant. In this assignment, you will be asked to solve this problem.

Task Description

Csereo is a serial cereal consumer. She reopens an old cereal box with a binary serial code M of length m. She realizes that the code is a link to a fantasy realm of cereal, and to unlock its power to foresee reality, she really needs to find instances of a code N of length n within the cereal serial code M. However, some characters in M have faded away and are unreadable. These are represented by `?` characters which can represent either `0` or `1`.

Csereo wants to find the number of occurrences of N within the code M. She considers an occurrence to be a substring $M[j..j+n-1]$ such that $M[j..j+n-1]$ is equal to N but with at most K characters replaced by `?`. Can you help her out?

Input

The input will contain three lines. The first line of input will contain a single integer K. The next line of input will contain string M and the last line will contain string N.

Output

Your program should output a single integer representing the number of matches of N in M where the number of `?` characters in the match do not exceed K.

Examples

Example 1	
Input	Output

`1`
`1?1?0?11??11`
`111`
	`4`

Explanation

Below are the occurrences of `111` in `1?1?0?11??11`:

- `1?1?0?11??11`. This is a valid match since only one wildcard is used.
- `1?1?0?11??11`. This is an invalid match since two wildcards are used.
- `1?1?0?11??11`. This is a valid match since only one wildcard is used.
- `1?1?0?11??11`. This is a valid match since only one wildcard is used.
- `1?1?0?11??11`. This is an invalid match since two wildcards are used.
- `1?1?0?11??11`. This is an invalid match since two wildcards are used.
- `1?1?0?11??11`. This is a valid match since only one wildcard is used.

Therefore the answer is 4 (i.e. `1?1?0?11??11`, `1?1?0?11??11`, `1?1?0?11??11`, `1?1?0?11??11`).

Example 2	
Input	Output

`4`
`??????????`
`1101`
	`7`

Explanation

Since at most 4 wildcards are allowed, any match of `1101` in `??????????` is valid. There are 7 matches in total, i.e. $\underline{\texttt{????}}\texttt{??????}$, $\texttt{?}\underline{\texttt{????}}\texttt{?????}$, …, $\texttt{??????}\underline{\texttt{????}}$.

Limits

For all tasks, $m \leq 2000000, n \leq m, K \leq 10$.

Tasks

Partial Task (1 mark)

For the partial task, there are no `?` characters. To obtain the full 1 mark, your algorithm should run in time complexity $O(m)$, where m is the length of the string M.

Full Task (1 mark)

For the full 1 mark, your algorithm should run in time complexity $O(m2^K)$ or better, where m is the length of the string M.

Hints

The Karp-Rabin algorithm solves string matching (without wildcards) in linear time using a rolling hash. Consider how you can modify Karp-Rabin to incorporate the setting with wildcards. The runtime requirement may provide a hint.

Submission Instructions

You will have to submit in **either Java or C++** to the submission portal at CodeCrunch (`codecrunch.comp.nus.edu.sg`) by **6th March 2022, 23:59**, after which the portal will automatically close. Please check that you have submitted to the **correct language**.

To get the full score for each task, your program needs to solve **ALL** testcases correctly. As this is a programming assignment, not only does your algorithm need to have a good (theoretical) time complexity, but you also need to have an efficient implementation. We also reserve the right to reject unintended solutions with less efficient time complexities.

Remark

There exists an entirely deterministic algorithm to solve the problem in $O(m2^K)$ time.