Sopa de letras (sopa)

Enunciado

Nately tiene una cadena infinita S formada por la repetición infinita de una palabra de longitud N. Por ejemplo, si la palabra es "primo", entonces $S =$ "primoprimoprimoprimo...".

Nately quiere jugar con Penguino un divertido juego, el cual será revelado en otra ronda de la olimpiada. Sin embargo, nota que Penguino le tiene fobia a algunas letras luego de su traumático paso por Ciudad Fritos. Tu tarea es ayudar a Nately a limpiar ciertas partes de esa cadena para que Penguino pueda jugar sin preocupaciones.

Penguino tiene fobia a ciertas letras, las cuales se te darán en otra cadena T de longitud M (cada letra aparece una sola vez en T). Estas letras representan los caracteres que Penguino no puede ver en la cadena, y por lo tanto deben ser eliminados de cualquier segmento que se quiera limpiar.

Para cada consulta, se te dará un rango $[A, B]$ que representa un substring de la cadena infinita S. Tu objetivo es determinar cuántas letras hay que eliminar del substring $S[A, B]$ para que no contenga ninguna de las letras a las que Penguino tiene fobia. Es decir, debes contar cuántas veces aparecen las terrorificas letras en ese segmento y reportar ese número como la cantidad mínima de eliminaciones necesarias para que Nately pueda jugar con Penguino.

No olvides que Nately tiene la extrana costumbre de enumerar las posiciones de una cadena empezando por el 1. Es decir, por ejemplo, en la cadena "primoprimoprimoprimo...", la posición 1 corresponde a la letra 'p', la posición 2 a la letra 'r', la posición 3 a la letra 'i', y así sucesivamente.

Ayuda a Nately a responder todas las consultas para que Penguino pueda disfrutar del juego sin preocuparse por sus fobias.

Para ello, deberás implementar dos funciones:

- `init(N, S, M, T)`: Inicializa la palabra base S (de longitud N) y la cadena de letras prohibidas T (de longitud M).
- `responder(A, B)`: Devuelve el número mínimo de letras que hay que eliminar del substring $S[A, B]$ para que no contenga ninguna letra de T.

Entrada

- Primera línea: dos enteros N, M
- Segunda línea: la palabra base S (de longitud N)
- Tercera línea: la cadena de letras prohibidas T (de longitud M)
- Cuarta línea: un entero Q, el número de consultas
- Siguientes Q líneas: dos enteros A, B para cada consulta

Recuerda que tu programa no debe leer de la entrada estándar, toda la lectura sera realizada por el archivo `grader.cpp` o `grader.py`. Tu solo debes modificar las funciones `init` y `responder`.

Salida

El grader se encargara de la salida, la salida consiste en imprimir un entero por cada consulta: el número mínimo de letras que hay que eliminar del substring $S[A, B]$ para que no contenga ninguna letra de T, tu no debes imprimir este valor, solo debes retornar el valor desde la función `responder`, el grader se encargara de imprimirlo.

Recuerda que no debes modificar el archivo `grader.cpp` o `grader.py`.

Ejemplos

Entrada	Salida

`6 2`
`friend`
`rn`
`4`
`3 7`
`3 8`
`1 6`
`1 81`
	`1`
	`2`
	`2`
	`32`

Límites

- $1 \leq N \leq 10^5$
- $1 \leq M \leq 26$
- $1 \leq Q \leq 10^5$
- $1 \leq A \leq B \leq 10^{18}$
- S y T contienen solo letras minúsculas del alfabeto inglés ('a'-'z')
- Todas las letras en T son distintas

Subtareas

Subtarea 1 (7 puntos)

- $N = 1$, es decir, la cadena infinita S está formada por un solo carácter repetido.

Subtarea 2 (32 puntos)

- $M = 1$, es decir, penguino tiene fobia a una sola letra.

Subtarea 3 (16 puntos)

- $1 \leq A \leq B \leq 1000$
- $1 \leq Q \leq 100$

Subtarea 4 (24 puntos)

- $A = B$, es decir, se consulta un solo carácter de la cadena infinita S.

Subtarea 5 (3 puntos)

- $M = 26$, es decir, penguino tiene fobia a todas las letras del alfabeto.

Subtarea 6 (18 puntos)

- Sin restricciones adicionales al problema original.