ADN Mutante

Grace es una bióloga trabajando para una compañía de bioinformática en Singapur. Como parte de su trabajo, ella analiza las secuencias de ADN de varios organismos. Una secuencia de ADN se define como una cadena de caracteres "A", "T" y "C". Note que en este problema las secuencias de ADN **no contienen el caracter "G".**

Definimos una mutación como una operación en la secuencia de ADN donde dos elementos de la secuencia son intercambiados. Por ejemplo, una sóla mutación puede cambiar "A**C**T**A**" en "A**A**T**C**" intercambiando los caracteres "A" y "C" marcados.

La distancia de mutación entre dos secuencias es el mínimo número de mutaciones requerida para transformar una secuencia en otra, o -1 si no es posible transformar una secuencia en otra utilizando mutaciones.

Grace está analizando dos secuencias de ADN a y b, ambas de n elementos con índices de 0 a $n-1$. Tu tarea es ayudar a Grace a contestar q preguntas de la forma: ¿cuál es la distancia de mutación entre la subcadena $a[x..y]$ y la subcadena $b[x..y]$?

Aquí, una subcadena $s[x..y]$ de una secuencia de ADN s se define como una secuencia de caracteres consecutivos de s cuyos índices van de x a y, inclusive. En otras palabras, $s[x..y]$ es la secuencia $s[x]s[x+1]\ ...\ s[y]$.

Detalles de implementación

Deberás implementar la siguiente función:

```
void init(string a, string b)
```

- a, b: cadenas de tamaño n, describiendo las dos secuencias de ADN a ser analizadas.
- Esta función es llamada exactamente una vez, antes de cualquier llamada a `get_distance`.

```
int get_distance(int x, int y)
```

- x, y: los índices de inicio y fin de las subcadenas a analizar.
- La función debe retornar la distancia de mutación entre las subcadenas $a[x..y]$ y $b[x..y]$.
- Esta función se llama exactamente q veces.

Ejemplo

Considere la siguiente llamada:

```
init("ATACAT", "ACTATA")
```

Supongamos que el calificador realiza la llamada `get_distance(1, 3)`. Esta llamada debe retornar la distancia de mutación entre $a[1..3]$ y $b[1..3]$, es decir, las secuencias "TAC" y "CTA". "TAC" puede ser transformada en "CTA" utilizando 2 mutaciones: **TA**C $\rightarrow$ **CA**T, seguida de C**AT** $\rightarrow$ C**TA**, y la transformación es imposible utilizando menos de 2 mutaciones.

Por lo tanto, la llamada deberá retornar 2.

Ahora supongamos que el calificador realiza la llamada `get_distance(4, 5)`. Esta llamada debe retornar la distancia de mutación entre las secuencias "AT" y "TA". "AT" puede ser transformada en "TA" utilizando una sóla mutación, y claramente al menos una mutación es requerida.

Por lo tanto, esta llamada debe retornar 1.

Finalmente, supongamos que el calificador realiza la llamada `get_distance(3, 5)`. Ya que **no existe** una forma en que la secuencia "CAT" sea transformada en "ATA" utilizando cualquier secuencia de mutaciones, esta llamada debe retornar -1.

Restricciones

- $1 \leq n, q \leq 100\ 000$
- $0 \leq x \leq y \leq n - 1$
- Cada caracter de a y b es uno de los caracteres "A", "T" o "C".

Subtareas

1. (21 puntos) $y - x \leq 2$
2. (22 puntos) $q \leq 500,\ y - x \leq 1000$, cada caracter de a y b es "A" o "T".
3. (13 puntos) a y b se forman utilizando únicamente los caracteres"A" o "T".
4. (28 puntos) $q \leq 500,\ y - x \leq 1000$
5. (16 puntos) Sin restricciones adicionales.

Calificador de ejemplo

El calificador de ejemplo lee la entrada en el siguiente formato:

- línea 1: $n\ \ q$
- línea 2: a
- línea 3: b
- línea $4 + i$ ($0 \leq i \leq q - 1$): $x\ \ y$ para la i-ésima llamada a `get_distance`.

El calificador de ejemplo imprime tus respuestas en el formato siguiente:

- línea $1 + i$ ($0 \leq i \leq q - 1$) : el valor de retorno de la i-ésima llamada a `get_distance`.