Mutating DNA

Grace je biolog koja radi u firmi za bioinformatiku u Singapuru. Kao dio svog posla, ona analizira DNK sekvence različitih organizama. DNK sekvenca je definisana kao string koji se sastoji od znakovi „A“, „T“ i „C“. Imajte na umu da u ovom zadatku DNK sekvence **ne sadrže znak "G".**

Definisaćemo mutaciju kao operaciju na jednoj DNK sekvenci gdje su dva elementa sekvence zamjenila mjesta. Na primer, pojedinačna mutacija može transformisati "A**C**T**A**" u "A**A**T**C**" zamjenom boldovanih znakova „A“ i „C“

Udaljenost mutacije između dvije sekvence je najmanji broj mutacija potrebnih da se transformiše jedna sekvenca u drugu ili -1 ako nije moguće transformisati jednu sekvencu u drugu korišćenjem mutacija.

Grejs analizira dvije DNK sekvence a i b, obje se sastoje od n elemenata sa indeksima od 0 do $n-1$. Vaš zadatak je da pomognete Grace da odgovori na q pitanja oblika: koja je udaljenost mutacije između podstringa $a[x..y]$ i podstringa $b[x..y]$? Ovde je podstring $s[x..y]$ DNK sekvence s je definisan kao sekvenca uzastopnih karaktera stringa s, čiji su indeksi x do y uključivo. Drugim riječima, $s[x..y]$ je sekvenca $s[x]s[x+1]\ldots s[y]$.

Detalji implementacije

Treba implementirati sledeću funkciju:

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

- a, b: stringovi dužine n, koji opisuju dvije DNK sekvence koje treba analizirati.
- Ova funkcija se poziva tačno jednom, prije bilo kojeg poziva `get_distance`.

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

- x, y: početni i kranji indeksi substringova koje treba analizirati.
- Funkcija treba da vrati udaljenost mutacije između podstringova $a[x..y]$ i $b[x..y]$.
- Funkcija se poziva tačno q puta.

Primjer

Razmotrite sledeći poziv:

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

Recimo da program za ocjenjivanje (grader) poziva `get_distance(1, 3)`. Ovaj poziv treba da vrati udaljenost mutacije između $a[1..3]$ i $b[1..3]$, odnosno sekvence "TAC" and "CTA". "TAC" se može transformisati u "CTA" pomoću 2 mutacije: T**A**C $\rightarrow$ **C**A**T**, nakon čega slijedi C**AT** $\rightarrow$ C**TA**, a transformacija je nemoguća sa manje od 2 mutacije.

Prema tome, ovaj poziv treba da vrati 2.

Recimo da program za ocjenjivanje (grader) poziva `get_distance(4, 5)`. Ovaj poziv treba da vrati udaljenost mutacije između "AT" and "TA". "AT" može biti transformisan u "TA" jedom mutacijom, očigledno je potrebna najmanje jedna mutacija.

Prema tome, ovaj poziv treba da vrati 1.

Na kraju, recimo da program za ocjenjivanje (grader) poziva `get_distance(3, 5)`. Budući da **nema načina** da se sekvenca "CAT" transformiše u "ATA" putem bilo kog niza mutacija, ovaj poziv treba da vrati -1.

Ograničenja

- $1 \leq n, q \leq 100\ 000$
- $0 \leq x \leq y \leq n - 1$
- Svaki znak od a i b jedan je od znakova "A", "T", i "C".

Podzadaci

1. (21 bod) $y - x \leq 2$
2. (22 boda) $q \leq 500,\ y - x \leq 1000$, svaki znak od a i b je ili "A" ili "T".
3. (13 bodova) svaki znak a i b je "A" ili "T".
4. (28 bodova) $q \leq 500,\ y - x \leq 1000$
5. (16 bodova) Nema dodatnih ograničenja.

Primjer programa za ocjenjivanje (Sample grader)

Program za ocjenjivanje (sample grader) čita ulaz u sledećem formatu:

- linija 1: n q
- linija 2: a
- linija 3: b
- linija $4 + i$ ($0 \leq i \leq q - 1$): x y za i-ti poziv `get_distance`.

Program za ocjenjivanje (sample grader) prikazuje odgovor u sledećem formatu:

- linija $1 + i$ ($0 \leq i \leq q - 1$) : vraća vrijednost i-tog poziva `get_distance`.