Mutējošā DNS

Greisa ir bioloģe, kas strādā kādā Singapūras bioinformātikas firmā. Viņas darbs ir saistīts ar dažādu organismu DNS virkņu analīzi. DNS virkni definē kā simbolu virkni, kas sastāv tikai no simboliem "A", "T" un "C". Ievērojiet, ka šajā uzdevumā DNS virknēs **nav simbola "G".**

Par mutāciju sauksim darbību ar DNS virkni, kur divi virknes elementi tiek samainīti vietām. Piemēram, viena mutācija var pārveidot "A**C**T**A**" par "A**A**T**C**", samainot vietām izceltos simbolus "A" un "C".

Divu virkņu mutāciju attālums ir mazākais mutāciju skaits, kāds nepieciešams, lai vienu virkni pārveidotu par otru, vai arī -1, ja vienu virkni pārveidot par otru ar mutāciju palīdzību nav iespējams.

Greisa analizē divas DNS virknes a un b, kur katrā no tām ir n simboli, kas indeksēti no 0 līdz $n-1$. Jūsu uzdevums ir palīdzēt Greisai atbildēt uz q šāda veida jautājumiem: kāds ir mutāciju attālums starp virkņu fragmentiem $a[x..y]$ un $b[x..y]$? Šeit DNS virknes s fragments $s[x..y]$ tiek definēts kā s secīgu simbolu ar indeksiem no x līdz y ieskaitot virkne. Citiem vārdiem, $s[x..y]$ ir virkne $s[x]s[x+1]\ldots s[y]$.

Realizācijas detaļas

Jums jārealizē šādas procedūras:

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

- a, b: divas analizējamās DNS virknes garumā n.
- Šī procedūra tiek izsaukta tieši vienu reizi un pirms tam, kad kaut vienu reizi tiek izsaukta procedūra `get_distance`.

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

- x, y: analizējamo fragmentu sākuma un beigu indeksi.
- Procedūrai jāatgriež mutāciju attālums starp fragmentiem $a[x..y]$ un $b[x..y]$.
- Šī procedūra tiek izsaukta tieši q reizes.

Piemērs

Aplūkosim šādu izsaukumu:

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

Pieņemsim, ka vērtētājs izsauc `get_distance(1, 3)`. Šim izsaukumam ir jāatgriež mutāciju attālums starp $a[1..3]$ un $b[1..3]$, tas ir, starp virknēm "TAC" un "CTA". "TAC" var pārveidot par "CTA" ar 2 mutāciju palīdzību: **TA**C $\rightarrow$ **CA**T, kam seko C**AT** $\rightarrow$ C**TA**, turklāt pārveidošana ar mazāk nekā 2 mutācijām nav iespējama.

Tādējādi šim izsaukumam ir jāatgriež 2.

Pieņemsim, ka vērtētājs izsauc `get_distance(4, 5)`. Šim izsaukumam ir jāatgriež mutāciju attālums starp "AT" un "TA". "AT" var pārveidot par "TA" ar vienas mutācijas palīdzību un, acīmredzami, vismaz viena mutācija ir nepieciešama.

Tādējādi šim izsaukumam ir jāatgriež 1.

Visbeidzot, pieņemsim, ka vērtētājs izsauc `get_distance(3, 5)`. Tā kā no virknes "CAT" ar mutāciju palīdzību **nekādi nav iespējams** iegūt virkni "ATA", šim izsaukumam ir jāatgriež -1.

Ierobežojumi

- $1 \leq n, q \leq 100\ 000$
- $0 \leq x \leq y \leq n - 1$
- Katrs virkņu a un b simbols ir vai nu "A", vai "T", vai "C".

Apakšuzdevumi

1. (21 punkts) $y - x \leq 2$
2. (22 punkti) $q \leq 500,\ y - x \leq 1000$, katrs a un b simbols ir vai nu "A", vai "T".
3. (13 punkti) katrs a un b simbols ir vai nu "A", vai "T".
4. (28 punkti) $q \leq 500,\ y - x \leq 1000$
5. (16 punkti) Bez papildu ierobežojumiem.

Paraugvērtētājs

Paraugvērtētājs datus ielasa šādā formātā:

- 1. rinda: $n\ \ q$
- 2. rinda: a
- 3. rinda: b
- ($4 + i$)-tā rinda ($0 \leq i \leq q - 1$): $x\ \ y$ i-tajam `get_distance` izsaukumam.

Paraugvērtētājs izvada atbildi šādā formātā:

- line $1 + i$ ($0 \leq i \leq q - 1$): i-tā `get_distance` izsaukuma atgriežamā vērtība.