# DNA Mutasyonu

Biyolog Grace Singapur'da bir biyoenformatik firmasında çalışıyor. Yaptığı işlerin bir parçası da farklı organizmaların DNA dizilimlerini analiz etmek. DNA dizilimi "A", "T", ve "C" karakterlerinden oluşan bir karakter dizisi olarak tanımlanmakta. Bu problemde tanımlanan DNA dizilimlerinde **"G" karakteri bulunmadığına dikkat ediniz**.

DNA dizilimlerinde iki elemanın yer değiştirmesini mutasyon işlemi olarak tanımlıyoruz. Örneğin, bir mutasyon, kalın karakterlerle belirtilmiş "A" ve "C" karakterlerinin yer değiştirilmesiyle, "A**C**T**A**" dizilimini "A**A**T**C**" dizilimine dönüştürebilir.

İki dizilim arasındaki mutasyon uzaklığı da bir dizilimi diğerine dönüştürmek için gerekli olan mutasyon sayısı olarak tanımlanmıştır. Eğer mutasyon kullanılarak böyle bir dönüşüm gerçekleştirilmesi mümkün değilse bu uzaklık $-1$ olacaktır.

Grace verilen $a$ ve $b$ dizilimlerinin analizini yapmaktadır, her iki dizi de $0$'dan $n-1$'e kadar indekslenmiş, $n$ adet elemandan oluşmaktadır. Sizin göreviniz $a[x..y]$ altdizisi ile $b[x..y]$ altdizisi arasındaki mutasyon uzunluğunun sorulduğu $q$ adet soruyu cevaplaması gereken Grace'e yardımcı olmanız. $s[x..y]$ olarak gösterilen bir altdizi tanımı, DNA dizilimi $s$ için, $x$ ile $y$ ($x$ ve $y$ dahil) arasında birbirini takip eden karakter dizisi olarak yapılmıştır. Başka bir deyişle, $s[x..y]$ dizisi $s[x]s[x+1]\ldots s[y]$ olacak şekilde tanımlanmıştır.

## Implementasyon Detayları

Aşağıda verilen fonksiyonları kodlamalısınız:

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

- $a$, $b$: Analiz edilecek olan iki DNA dizilimini tanımlayan, $n$ uzunluğunda string dizisi
- Bu fonksiyon `get_distance` fonksiyonlarından önce tam olarak bir defa çalıştırılacak

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

- $x$, $y$: Analiz edilecek olan altdizi için başlangıç ve bitiş indeksleri
- Bu fonksiyon verilen iki altdizi, $a[x..y]$ ve $b[x..y]$, için mutasyon uzunluğunu döndürecektir
- Bu fonksiyon tam olarak $q$ defa çağırılacaktır

## Örnekler

Aşağıda verilen fonksiyon çağrısı yapıldığında:

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

Grader `get_distance(1, 3)` fonksiyonunu çağırmış olsun. Bu fonksiyon $a[1..3]$ ve $b[1..3]$ altdizileri, "TAC" ve "CTA" olacaktır, arasındaki mutasyon uzunluğunu döndürecektir. "TAC" $2$ mutasyon ile "CTA" dizilimine dönüştürülebilir: **TA**C $\rightarrow$ **C**A**T**, ardından C**AT** $\rightarrow$ C**TA**, ve bu dönüşüm $2$ mutasyondan az mutasyonla yapılamaz.

Dolayısıyla, bu fonksiyon $2$ değerini döndürmelidir.

Grader `get_distance(4, 5)` fonksiyonunu çağırmış olsun. Bu fonksiyon "AT" ve "TA" altdizileri arasındaki mutasyon uzunluğunu döndürmelidir. "AT" tek bir mutasyonla "TA" dizilimine dönüştürülebilir, ve sadece bir mutasyon gereklidir.

Dolayısıyla, bu fonksiyon $1$ değerini döndürmelidir.

Son olarak, grader `get_distance(3, 5)` fonksiyonunu çağırmış olsun. Bu durumda "CAT" dizisinin "ATA" dizisine çevirebilecek herhangi bir mutasyon dizisi **olamayacağı** için bu fonksiyon $-1$ değerini döndürmelidir.

## Kısıtlar

- $1 \leq n, q \leq 100\ 000$
- $0 \leq x \leq y \leq n-1$
- $a$ ve $b$ dizilerindeki her bir karakter "A", "T", ve "C" karakterlerinden biridir.

## Altgörevler

1. (21 puan) $y - x \leq 2$
2. (22 puan) $q \leq 500,\ y - x \leq 1000,\ a$ ve $b$ dizilimlerindeki her karakter "A" veya "T" karakterlerinden oluşmaktadır.
3. (13 puan) $a$ ve $b$ dizilimlerindeki her karakter "A" veya "T" karakterlerinden oluşmaktadır.
4. (28 puan) $q \leq 500,\ y - x \leq 1000$
5. (16 puan) Ek kısıt bulunmamaktadır.

## Örnek grader

Örnek grader girdileri aşağıdaki formatta okuyacaktır:

- satır $1$: $n\ \ q$
- satır $2$: $a$
- satır $3$: $b$
- satır $4 + i$ ( $0 \leq i \leq q - 1$): $i$'ninci `get_distance` çağrısı için $x\ \ y$ değerleri.

Örnek grader cevaplarınızı aşağıdaki formatta yazacaktır:

- satır $1 + i$ ( $0 \leq i \leq q - 1$) : $i$'ninci `get_distance` çağrısı için döndürülen değer