Sekwencjonowanie BNA (A)

Limit pamięci: 256 MB Limit czasu: 10.00 s

To zadanie jest interaktywne. Po wypisaniu każdego wiersza należy opróżnić bufor. W C++ możesz użyć **cout « flush**, w Java – **System.out.flush()**, w Python – **sys.stdout.flush()**. Pamiętaj o wypisaniu białego znaku podczas opróżniania bufora. Należy ściśle trzymać się protokołu opisanego w rozdziale *Protokół interakcji* – niewczytanie wiadomości wysłanej przez interaktor skutkuje werdyktem *wrong answer*.

Rozpoczęte zostały prace nad opracowaniem leku na bitozę – chorobę genetyczną prześladującą mieszkańców Bitocji od paru dobrych wieków. Ostatnio dokonano pewnego postępu w badaniach. Bitoccy naukowcy podejrzewają, że udało im się wyizolować segment BNA odpowiedzialny za wywoływanie choroby.

BNA (kwas bitrybonukleinowy) jest organicznym związkiem chemicznym, który koduje informację genetyczną. BNA może być opisane jako ciąg kodujących go kwasów bitowych: zerozyny (“`0`”) oraz jedyniny (“`1`”). Aktualnym problemem w kontynuowaniu badań jest poznanie ciągu kwasów bitowych segmentu BNA, który odpowiada za bitozę.

Bajtazar – jako osoba odpowiedzialna za nadzór nad badaniami – poprosił ciebie, znajomego bioinformatyka, o napisanie programu komputerowego, który odczyta sekwencję BNA kodującego bitozę. Aby ci to umożliwić, Bajtazar udostępnił ci technologię pozwalającą na przeprowadzenie testów stwierdzających, czy sekwencja syntetycznie wytworzonego BNA znajduje się jako **spójny podciąg** sekwencji BNA bitozy. Niestety, przeprowadzenie pojedynczego testu może trwać nawet parę godzin, dlatego zostałeś poproszony o to, aby twój program wykonał jak najmniej testów.

Interakcja

Na początku w pierwszym wierszu wejścia znajdzie się liczba całkowita N, oznaczająca długość sekwencji BNA bitozy.

Następnie należy przeprowadzić pewną (być może zerową) liczbę testów przy pomocy zapytań. Zapytania mają postać `? A`, gdzie `A` jest ciągiem znaków `0` oraz `1` oznaczającym sekwencję syntetycznego BNA do zastosowania w teście. Odpowiedzią na zapytanie jest `1`, jeśli sekwencja syntetycznego BNA znajduje się jako spójny podciąg w sekwencji BNA odpowiedzialnej za bitozę; w przeciwnym wypadku interaktor wypisze `0`.

Po wykonaniu wszystkich zapytań, należy wypisać sekwencję BNA bitozy `B` w formacie: `! B`.

Wypisanie sekwencji BNA bitozy nie wlicza się do liczby zapytań.

Interaktor jest adaptacyjny, oznacza to, że sekwencja BNA bitozy **nie musi być ustalona z góry** i może być dobierana na bieżąco w trakcie działania programu, o ile jej zmiana nie zaprzeczy wynikom wykonanych do tej pory zapytań.

Punktacja

We wszystkich testach zachodzi: $1 \leq N \leq 10\,000$.

Liczba punktów **zależy od liczby zadanych zapytań**.

Oznaczmy przez Q maksymalną liczbę zapytań zadanych przez program. Wówczas, liczba punktów zależy w następujący sposób od Q:

Próg Q	Liczba punktów
$30\,000$	10
$20\,000$	20
$15\,000$	40
$11\,000$	60
$10\,200$	70
$10\,035$	85
$10\,025$	90
$10\,015$	100

Jeśli $Q \geq 30\,000$, to otrzymasz 0 punktów za to zadanie.

W przeciwnym wypadku, gdy $Q \leq 10\,015$, otrzymasz 100 punktów.

W pozostałych przypadkach, jeśli Q jest równe któremuś z powyższych progów w tabeli, otrzymasz przypisaną mu liczbę punktów.

W przeciwnym razie Q znajduje się pomiędzy dwoma progami i otrzymasz liczbę punktów wyznaczoną liniową interpolacją pomiędzy nimi, z zaokrągleniem w dół.

Przykładowa interakcja

Wejście	Wyjście
4	
	? 0
1	
	? 00
0	
	? 1
1	
	? 11
0	
	? 0101
0	
	? 00000
0	
	! 1010

Powyższa interakcja została przeprowadzona poprawnie. W szczególności zapytanie o długości większej niż N jest dopuszczalne. W powyższej interakcji Q jest równe 6, a więc $Q \leq 10\,015$, co oznacza, że za powyższy test zostanie przydzielone 100 punktów.

Narzędzia do testowania lokalnego

Zostaną udostępnione testy przykładowe i program **interaktor.cpp** oraz przykładowe błędne rozwiązanie o poprawnym schemacie komunikacji **wa.cpp**. Żeby uruchomić rozwiązanie na teście, należy skompilować rozwiązanie i interaktorkę, a następnie użyć komendy:

`[plik wykonywalny z interaktor.cpp] [test] [plik wykonywalny z rozwiązaniem]`

Na przykład: `./interaktor 0 ./wa`