Zadanie: OSC
Oszczędności

polish

ONTAK, dzień 4. Dostępna pamięć: 512 MB. Limit czasu: 2 s. *05.07.2024*

Twój samolot osiągnął właśnie wysokość przelotową, pogoda jest piękna. Być może kariera pilota nie była takim złym pomysłem, chociaż chodzą słuchy, że producent samolotów o nazwie zaczynającej się na B* wprowadził ostatnio nowe oszczędności w procesie produkcji i obsługi floty. Ale na razie nic się nie dzieje... zaraz, co to za dziwny dźwięk? Aha, to producent wprowadził nowy rodzaj alarmu, zastępując tradycyjny elektroniczny brzęczyk zwykłym dzwonkiem, który wygląda na ściągnięty z jakiejś owcy...

W kokpicie zapaliły się dwie lampki alarmowe o numerach A i B. Procedura w razie awarii jest następująca: musisz zaglądnąć do podręcznika na strony numer A i B, przeczytać z nich opisy procedur i wykonać tę z nich, której opis jest późniejszy w porządku alfabetycznym. Opisy procedur **są parami różne**.

Problem polega na tym, że podręcznik został na lotnisku – w ramach oszczędności produkowany jest tylko jeden egzemplarz, który trzyma wieża kontrolna. W teorii wystarczy więc zapytać wieży, która odczyta Ci odpowiednie procedury. Cały problem w tym, że pierwszą rzeczą, która właśnie wysiadła, był Twój nadajnik radiowy. Możesz komunikować się z wieżą wyłącznie alfabetem Morse'a, wysyłając im dokładnie 9 bitów informacji (na więcej nie starczy Ci mocy). Wieża może odesłać Ci więcej bitów, ale zgodnie z polityką oszczędnościową firmy koszt takiej komunikacji zostanie potrącony z Twojej wypłaty. Spróbuj więc przeprowadzić całą komunikację tak, żeby tych bitów było jak najmniej.

Aha, jeszcze jedna kwestia – co należy powiedzieć pasażerom? Nowa, oszczędna zasada mówi, żeby spojrzeć na pierwszy znak od lewej, na którym różnią się ciągi numer A i B. Jeśli jest on na miejscu parzystym $(0, 2, 4, \ldots)$, należy poinformować pasażerów o sytuacji, jeśli na nieparzystym $(1, 3, 5 \ldots)$, nie.

Czy możliwa jest komunikacja między samolotem a wieżą, która za pomocą przekazania 9 bitów w jedną stronę (samolot $\rightarrow$ wieża) i jak najmniejszej ich liczby w drugą stronę (wieża $\rightarrow$ samolot) rozstrzygnie, czy ciąg znaków pod numerem A w książce będącej w posiadaniu wieży jest alfabetycznie mniejszy czy większy od ciągu pod numerem B, oraz poda parzystość miejsca, na którym te ciągi się różnią? Na początku tylko samolot zna A i B, zaś tylko wieża posiada książkę.

Interakcja

To jest zadanie interaktywne. Dostarczamy Ci szablon programu `osc.cpp`, który zawiera funkcję `main()` (odradzamy jakiekolwiek modyfikacje tej funkcji), oraz dwie funkcje `samolot()` i `wieza()`, których ciało musisz uzupełnić. Pierwsza z nich będzie odpowiadała za komunikację samolotu, druga – za komunikację wieży. **Twój program zostanie uruchomiony więcej niż raz, w różnych rolach** – w szczególności jedna instancja nigdy nie będzie działała jednocześnie jako samolot i jako wieża.

Funkcja `samolot()`. Do komunikacji samolotu przeznaczona jest funkcja `void samolot()`. Powinna ona wykonać kolejno następujące czynności:

- Wczytać ze standardowego wejścia jeden wiersz zawierający pięć liczb całkowitych T, N, M, A, B ($2 \leq N \leq 1\,000$, $1 \leq M \leq 1\,000$, $0 \leq A, B \leq N - 1$, $A \neq B$) – numer podzadania, całkowitą liczbę procedur w książce, długość opisu każdej z procedur, oraz numery A i B procedur, które trzeba ze sobą porównać;
- Wypisać na standardowe wyjście wiersz zawierający ciąg **dokładnie** 9 bitów (cyfr `0` i `1`);
- Odczytać ze standardowego wejścia odpowiedź wieży, która będzie jednym wierszem, również zawierającym ciąg znaków `0` i `1`;
- Wypisać na standardowe wyjście wiersz zawierający dwa znaki oddzielone spacją: pierwszym powinno być `A` lub `B`, określający który ciąg znaków jest później w porządku alfabetycznym, zaś drugim znakiem `0` lub `1` – wypisz `0`, jeśli pierwszy od lewej znak różniący między sobą A i B jest na pozycji parzystej (numerację zaczynamy od 0), zaś `1`, jeśli jest na pozycji nieparzystej.

Funkcja `wieza()`. Funkcja `void wieza()` odpowiada za komunikację wieży. Jej zadaniem jest:

- Wczytać ze standardowego wejścia wiersz zawierający dwie liczby: numer T podzadania, liczbę N procedur w książce, oraz liczbę M, która jest wspólną długością wszystkich procedur;

*Ze względu na możliwość sporu prawnego nie możemy podać nazwy firmy, ale możemy zacytować jej motto: *When one door closes, another door opens.*

- Wczytać kolejnych N wierszy zawierających opisy procedur, z których każdy jest ciągiem M znaków `0` i `1`;
- Wczytać ze standardowego wejścia ciąg 9 bitów, który prześle jej samolot;
- Wypisać na standardowe wyjście ciąg przeznaczony dla samolotu, składający się ze znaków `0` i `1`. Od długości ciągu zależy liczba punktów (sekcja *Ocenianie*).

Po każdym zapytaniu i po każdej odpowiedzi należy wywołać polecenie `cout.flush()`, lub `fflush(stdout)` jeżeli używasz `printf`. Należy przestrzegać podanego wyżej formatu wejścia i wyjścia, łącznie z białymi znakami i podziałem na wiersze – w przeciwnym przypadku program może zakończyć się werdyktem *Przekroczenie limitu czasu* zamiast *Błędna odpowiedź*.

Przykładowa interakcja

Sprawdzarka do samolotu	**Samolot do sprawdzarki**	**Sprawdzarka do wieży**	**Wieża do sprawdzarki**	**Wyjaśnienie**
0 4 3 3 1				$T = 0, N = 4, M = 3, A = 3, B = 1$
	111000001			Samolot koduje informacje o A oraz B.
		0 4 3 111 101 010 011 111000001		$T = 0, N = 4, M = 3$. Pierwsza instrukcja. Druga instrukcja. Trzecia instrukcja. Czwarta instrukcja. Wieża dostaje informacje od samolotu.
			011101	Wieża wysyła informacje do samolotu przez sprawdzarkę. Na podstawie długości tego ciągu przydzielana będzie liczba punktów za test.
011101				Samolot dostaje odpowiedź od wieży.
	B 0			Samolot wypisuje odpowiedź – liczba na pozycji B, czyli 111 jest późniejsza alfabetycznie od liczby A, czyli 011.

Testy przykładowe:

`0a:` $T = 0$, $N = 4$, $M = 3$, $A = 3$, $B = 1$, instrukcje jak w przykładowej interakcji.

`0b:` $T = 0$, $N = 2$, $M = 1$, $A = 0$, $B = 1$, ciągi instrukcji to 0 oraz 1.

Ocenianie

Oznaczmy liczbę bitów przesłanych przez wieżę przez L. Wówczas:

- liczba punktów za pierwsze podzadanie wynosi 11 jeśli $L \leq 1\,000\,000$ i 0 w przeciwnym przypadku
- liczba punktów za drugie podzadanie wynosi 89 jeśli $L \leq 500$ oraz $89 \cdot \frac{500}{L}$ w przeciwnym przypadku. W szczególności, jeśli $L = 2000$, to Twój program dostanie 25% punktów za to podzadanie.

Dodatkowo, jeśli Twój program wypisze poprawnie tylko pierwszą liczbę, liczba punktów za test uzyskana według poniższego algorytmu zostanie przemnożona przez 0.4.

Eksperymenty

W sekcji *Pliki* dostępna jest przykładowa interaktorka `oscsoc.cpp`. Może się ona nieznacznie różnić od tej używanej do sprawdzania rozwiązań. Należy ją odpalać komendą:

```
python3 run.py 2 oscsoc [rozwiazanie] [test]
```

przy czym plik `oscsoc.cpp` oraz rozwiązanie muszą być skompilowane.