Jaskinia

WWI 2024 – grupa 3
Dzień 3 – 17 sierpnia 2024

Kod zadania: **jas**
Limit pamięci: **64 MiB**

Twój znajomy, który jest architektem, zaprosił Cię na otwarcie nowego parku rozrywki w górach. Jedną z głównych atrakcji jest korytarz w jaskini, który jest poprzedzielany n ($1 \leq n \leq 5000$) drzwiami. Właśnie wchodzisz do tej jaskini, aby ją obejrzeć i widzisz swojego znajomego, który siedzi z jakimiś tabliczkami starając się je poukładać. Okazuje się, że przed wejściem do jaskini jest tabliczka z n włącznikami, które służą do otwierania drzwi. Niestety został jeden dzień do otwarcia, a Twój znajomy zgubił plany jaskini i nie ma pojęcia, który włącznik jest do których drzwi. Zapamiętał jedynie, że każdy włącznik otwiera dokładnie jedne drzwi. Z uwagi na to, że Twój znajomy jest też bardzo ekscentrycznym architektem, to włącznik niekoniecznie musi być przełączony do góry, aby odpowiadające mu drzwi były otwarte, a w dodatku kolejność drzwi odpowiadających kolejnym włącznikom może być zupełnie dowolna.

Architekt zupełnie już stracił głowę do tych tabliczek. Czy pomożesz mu dopasować przełączniki do drzwi i ich ustawienie tak, aby wszystkie drzwi były otwarte? Jako, że macie bardzo mało czasu (w końcu otwarcie parku jest już jutro), to nie będziecie dzwonić po elektryka, który sprawdzi jak otworzyć wszystkie drzwi. Jedyne co wam pozostało to, aby wypróbować dane ustawienie włączników, przejść się po jaskini i zobaczyć, które drzwi są zamknięte jako pierwsze. Czas leci, więc na taki spacer możecie sobie pozwolić co najwyżej 70000 razy. Powodzenia!

Biblioteka

Jest to zadanie interaktywne, to znaczy Twój program będzie porozumiewał się z biblioteką. Aby użyć biblioteki, należy załączyć nagłówek `#include "jas.h"`. Biblioteka udostępnia następujące funkcje:

- `int init()` – funkcja ta powinna zostać wywoływana dokładnie raz, na początku działania programu. Podaje ona liczbę drzwi w jaskini i bez jej wywołania program nie będzie reagował na inne polecenia.
- `int wyprobuj(bool U[])` – funkcja ta sprawdza aktualne ustawienie przełączników i informuje, które drzwi jako pierwsze, idąc od początku korytarza, będą zamknięte. Jeśli wszystkie drzwi są otwarte to funkcja zróci -1. Zarówno drzwi jak i przełączniki są numerowane kolejnymi liczbami całkowitymi od 0, tzn. $0, 1, \ldots, n-1$. Funkcja `wyprobuj` zakłada, że `U` ma dokładnie n elementów i wartość `true` na pozycji i oznacza, że i-ty przełącznik jest w pozycji „do góry".
- `odpowiedz(bool U[], int P[])` – funkcja ta zleca sprawdzenie, czy wszystkie przełączniki są od- powiednio ustawione i czy są poprawnie przypisane do drzwi. Powinna być ona wywołana tylko raz, a po jej wykonaniu program powinien się zakończyć. i-ty element `U` powienien zawierać wartość `true` wtedy i tylko wtedy, gdy przełączenie i-tego przełącznika do góry spowoduje otwarcie odpowiadających mu drzwi. i-ty element `P` powinien zawierać indeks odpowiadających i-temu przełącznikowi drzwi.

Kompilacja na swoim komputerze

Do zadania dołączono przykładową biblioteczkę przydatną przy testowaniu rozwiązania. Pliki z archiwum `jasdlazaw.zip` dostepnego w zakładce "Pliki" należy wypakować do folderu z kodem źródłowym programu.
Aby program się skompilował należy załączyć nagłówek `#include "jas.h"`.
Program należy skompilować razem z biblioteką `jaslib.cc`. Można to zrobić za pomocą polecenia: `g++ twojprogram.cpp jaslib.cc -o twojprogram`.

Wyjście

Twój program nie powinien pisać na standardowe wyjście (stdout) ani czytać ze standardowego wejścia (stdin). Dozwolone jest pisanie na standardowe wyjście diagnostyczne (stderr), lecz pamiętaj, że zabiera to cenny czas.

Przykład

Funkcja	Wynik	Opis
`init()`	4	Dostajemy liczbę drzwi w jaskini
`wyprobuj([1, 0, 1, 1])`	1	Wypróbowujemy kombinację, gdzie pierwszy przełącznik jest do góry, drugi do dołu, trzeci do góry i czwarty do góry.
`wyprobuj([0, 1, 1, 0])`	0	Wypróbowujemy kombinację, gdzie pierwszy przełącznik jest do dołu, drugi do góry, trzeci do góry i czwarty do dołu.
`wyprobuj([1, 0, 1, 0])`	-1	Wypróbowujemy kombinację, gdzie pierwszy przełącznik jest do góry, drugi do dołu, trzeci do góry i czwarty do dołu. Wynik to -1 zatem wszystkie drzwi są otwarte
`odpowiedz([1, 0, 1, 0], [2, 0, 3, 1])`	`Program kończy się`	Sprawdzamy czy nasza kombinacja jest poprawna i trzymamy kciuki, żeby tak było. :)

Ocenianie

Żeby program dostał jakiekolwiek punkty, musi on działać zgodnie z wymaganiami, czyli jako pierwszą wywołać funkcję `init`, jako ostatnią `odpowiedz` i każdą z nich dokładnie raz. Jeśli powyższe warunki zostaną spełnione, to program jest oceniany w następujący sposób. Niech k oznacza liczbę wywołań funkcji `wyprobuj`.

1. Jeśli $k \leq 70000$ i mieści się w połowie czasu dla danego testu/grupy testów, to program dostanie maksymalną liczbę punktów przewidzianą za dany test/grupę testów.
2. Jeśli $k \leq 70000$ i program nie mieści się w połowie czasu dla danego testu/grupy testów (ale nie przekracza limitu czasu), to program dostanie liczbę punktów przewidzianą za dany test/grupę testów pomniejszoną liniowo w zależności od przekroczonego czasu.
3. Jeśli $k > 70000$ lub zostanie przekroczony limit czasu, to program dostanie 0 punktów za dany test/grupę testów.

Dodatkowo znane są takie podzadania:

Ocenianie

Podzadanie	Ograniczenia	Limit czasu	Liczba punktów
1	Dla każdego i przełącznik i jest podłączony do drzwi i. Twoim zadaniem jest wyznaczenie ustawienia otwierającego drzwi.	5 s	12
2	Ustawienie wszystkich przełączników w dół powoduje otwarcie wszystkich drzwi. Twoim zadaniem jest wyznaczenie połączeń między przełącznikami a drzwiami	5 s	13
3	$1 \leq n \leq 100$	5 s	21
4	$1 \leq n \leq 2000$	5 s	30
5	$1 \leq n \leq 5000$	5 s	24