Оперираща машина

Това е интерактивна задача. Има скрита пермутация P с големина N, съдържаща всички цели числа в интервала $[0, N)$, и целочислена променлива X, която в началото е със стойност 0.

Разполагате с машина с N бутона. Вие ще взаимодействате с журито като натискате бутони. Когато натиснете i-тия бутон, то машината ще добави $2^{P[i]}$ към X и след това ще върне **popcount** на X (броят на 1-ците в двоичното представяне на X).

Вашата задача е да определите скритата пермутация P.

Бележка: Скритата пермутация е фиксирана и не се променя в зависимост от вашите заявки.

Детайли по имплементация

- Включете хедърния файл `machine.h`.
- Имплементирайте функцията със следния формат: `vector < int > find_permutation(int N);` - тя трябва да върне вектор с големина N, задаващ скритата пермутация P.
- Може да използвате функцията със следния формат: `int press_button(int position);` - тя приема целочислен параметър `position` в интервала $[0, N)$, задаващ индекса на бутон, който натискате. Функцията ще върне положително цяло число, което задава popcount-а на X след натискане на бутона.

Оценяване

Нека T е броят на извикванията на `press_button`:

- Ако $T \leq Q$ и сте спазили всички изисквания по-рано, то ще получите всички точки за подзадачата (погледнете таблицата по-долу за Q).
- В противен случай ще се счита, че имате Wrong Answer.

Подзадачи

Подзадача	Точки	Допълнителни ограничения
1	8	$N = 2, Q = 400\,000$
2	15	$N \leq 10, Q = 400\,000$
3	32	$N \leq 500, Q = 400\,000$
4	25	$N \leq 3\,000, Q = 400\,000$
5	20	$N \leq 3\,000, Q = 250\,000$

Локален грейдър

Локалният грейдър чете входа в следния формат:

Вход 1: N
Вход 2: $P[0]$, $P[1]$, ..., $P[N-1]$

Тук P е пермутация с големина N, описваща скритата пермутация, която вашата програма трябва да познае.

Преди извикването на `find_permutation`, локалният грейдър ще провери дали векторът P е пермутация. Ако това не е така, то ще се отпечата съобщение "Vector `P` is not a permutation" и програмата ще завърши.

Ако локалният грейдър засече нередност (protocol violation на системата), то изходът на локалния грейдър ще е `Protocol Violation: <MSG>`, където `<MSG>` ще е едно от следните съобщения за грешка:

- **Invalid guess**: ако сте направили извикване на функцията `press_button`, при което индексът не е валиден, т.е. опитвате се да натиснете бутон L и L е по-малко от 0 или по-голямо или равно на N (големината на пермутацията).
- **Invalid size**: ако големината на върнатата пермутация не е равна на големината на пермутацията P.
- **Not a permutation**: ако върнатият вектор не е пермутация на числата от интервала $[0, N)$.

В противен случай, нека A е пермутацията, която Вие сте върнали. Тогава ще получите:

- **Ред 1**: **Wrong Answer**, ако A не съвпада с P, а в противен случай **Accepted**.
- **Ред 2**: $A[0]$, $A[1]$, ..., $A[N-1]$
- **Ред 3**: броят на извикванията на `press_button`.

Пример

Нека разгледаме случая, при който $N = 5$ и скритата пермутация е $P = [4, 2, 0, 1, 3]$. Функцията `find_permutation` се извиква по следния начин:

```
find_permutation(5)
```

Нека приемем, че тази функция прави следните извиквания на `press_button`:

Извикване	Стойност на X	Върната стойност
`press_button(3)`	2	1
`press_button(2)`	3	2
`press_button(0)`	19	3
`press_button(2)`	20	2

Стойността на X по време на изпълнението е неизвестна за нас; ние можем само да използваме върнатите стойности от всяко извикване, което представлява **popcount**-а на текущия X.

След като намерим скритата пермутация (използвайки няколко, възможо е и нула, извиквания на `press_button`), трябва да върнем тази пермутация.

За този пример, ще получим точки само ако върнем $[4, 2, 0, 1, 3]$ и в зависимост от броя на извикванията на функцията `press_button`.