Więźniowie

W więzieniu siedzi 500 więźniów. Pewnego dnia strażnik daje im szansę wcześniejszego zwolnienia. W specjalnym pokoju umieszcza dwie torby: A i B. Każda torba zawiera monety. W każdej torbie jest ich pomiędzy 1 a N włącznie. Liczba monet w torbie A jest inna niż liczba monet w torbie B. Strażnik stawia przed nimi wyzwanie. Polega ono na tym, że muszą ustalić, w której torbie jest mniejsza liczba monet.

W pokoju poza torbami z monetami jest tablica. Na tablicy zawsze jest zapisana dokładnie jedna liczba. Początkowo jest ona równa 0.

Strażnik wpuszcza więźniów do pokoju pojedynczo. Więzień, który wchodzi do pokoju nie wie, ani którzy więźniowie już z w nim byli, ani nawet ilu wcześniej więźniów odwiedziło ten pokój. Po wejściu do pokoju więzień czyta liczbę zapisaną na tablicy. Po przeczytaniu jej wybiera jedną z toreb A lub B. Następnie **otwiera** torbę, przelicza monety, dowiadując się ile monet ona zawiera. Potem więzień wykonuje jedną z dwóch **akcji**:

- Nadpisuje liczbę zastaną na tablicy wybraną nieujemną liczbą całkowitą i opuszcza pokój. Więzień może zarówno zmienić jak i zachować zastaną liczbę. Po tej akcji strażnik wpuszcza kolejnych więźniów (chyba, że każdy z 500 więźniów już był w pokoju).
- Wskazuje, która z dwóch toreb zawiera mniej monet. Ta akcja natychmiast kończy wyzwanie.

Strażnik nigdy nie wpuści tego samego więźnia dwa razy do pokoju.

Więźniowie wygrywają, jeśli którykolwiek z nich poprawnie wskaże torbę z mniejszą liczbą monet. Przegrywają, jeśli którykolwiek z nich wskaże błędną torbę lub jeśli żaden z 500 więźniów nie wskaże żadnej torby.

Przed startem wyzwania więźniowie zbierają się w świetlicy i ustalają wspólną **strategię** składającą się z trzech kroków

- Ustalają dodatnią liczbę x, która jest największą liczbą, jaką można zapisać na tablicy.
- Ustalają dla każdej liczby i zapisanej na tablicy ($0 \leq i \leq x$), która torba powinna być otwarta przez więźnia, który tę liczbę widzi na tablicy w momencie wejścia.
- Ustalają, jaką akcję powinien wykonać więzień po przeliczeniu monet w wybranej torbie. Konkretnie, dla każdej liczby i zapisanej na tablicy ($0 \leq i \leq x$) i każdej liczby monet j znalezionych w otwartej torbie ($1 \leq j \leq N$), ustalają
  - jaką liczbę między 0 a x włącznie powinien więzień zapisać na tablicy, albo
  - która torba powinna być wskazana jako mająca mniej monet

Gy więźniom się powiedzie, strażnik zwolni ich po dodatkowych x dniach.

Twoim zadaniem jest zaprojektowanie strategii dla więźniów, która zapewni im sukces (niezależnie od liczby monet w torbach A i B). Wartość Twojej strategii zależy od x (zobacz rozdział Podzadania, aby poznać szczegóły).

Szczegóły implementacyjne

Powinieneś napisać następującą funkcję

```
int[][] devise_strategy(int N)
```

- N: maksymalna liczba monet w każdej z toreb.
- Wartością tej funkcji powinna być tablica s tablic zawierających $N+1$ liczb całkowitych określających Twoją strategię.

Wartość x to o jeden mniej niż liczba elementów tablicy s. Dla każdego i takiego, że $0 \leq i \leq x$, tablica $s[i]$ określa, co więzień powinien zrobić, jeśli wchodząc do pokoju zastanie na tablicy liczbę i.

1. Wartość $s[i][0]$ równa 0 oznacza, że więzień powinien zajrzeć do torby A. Wartość $s[i][0]$ równa 1 oznacza, że więzień powinien zajrzeć do torby B.
2. Niech j będzie liczbą monet w wybranej torbie. Więzień powinien teraz wykonać następującą akcję:
    - Jeśli wartość $s[i][j]$ jest równa -1, więzeń powinien wskazać torbę A jako zawierającą mniej monet
    - Jeśli wartość $s[i][j]$ jest równa -2, więzeń powinien wskazać torbę B jako zawierającą mniej monet
    - Jeśli wartość $s[i][j]$ jest nieujemna, to więzień powinien napisać ją na tablicy. Zwróć uwagę na to, że $s[i][j]$ nie może przekroczyć x.

- Ta funkcja będzie wywoływana tylko raz.

Przykład

Rozważ następujące wywołanie:

```
devise_strategy(3)
```

Niech v oznacza liczbę, którą więzień widzi na tablicy przy wejściu do pokoju. Jedną ze strategii jest:

- Jeśli $v = 0$ (być może jest to wartość początkowa), sprawdź torbę A.
    - Jeśli zawiera 1 monetę, określ A jako tę, która zawiera mniej monet.
    - Jeśli zawiera 3 monety, określ B jako tę, która zawiera mniej monet.

    - Jeśli zawiera 2 monety, zapisz liczbę 1 na tablicy (nadpisując 0).
- Jeśli $v = 1$, sprawdź torbę B.
    - Jeśli zawiera 1 monetę, określ B jako tę, która zawiera mniej monet.
    - Jeśli zawiera 3 monety, określ A jako tę, która zawiera mniej monet.
    - Jeśli zawiera 2 monety, zapisz 0 na tablicy (nadpisując 1). Zwróć uwagę, że ten przypadek nie może się zdarzyć, gdyż wtedy obie torby zawierałyby po dwie monety, co jest niedopuszczalne. . Aby zakomunikować tę strategię, funkcja powinna przekazać tablicę zawierającą wartości `[[0, -1, 1, -2], [1, -2, 0, -1]]`. Długość tablicy, to 2, więc dla takiej wartości wynikiem jest x równe $2 - 1 = 1$.

Ograniczenia

- $2 \leq N \leq 5000$

Podzadania

1. (5 punktów) $N \leq 500$, wartość x nie może przekraczać 500.
2. (5 punktów) $N \leq 500$, wartość x nie może przekraczać 70.
3. (90 punktów), wartość x nie może przekraczać 60.

Jeśli w którykolwiek z testów tablica przekazana przez `devise_strategy` nie odpowiada poprawnej strategii, to wynik Twojego rozwiązania dla tego podzadania będzie równy 0.

W podzadaniu 3 możesz uzyskać wynik częściowy. Niech m będzie największą liczbą x dla wszystkich tablic przekazanych przez wystkie przypadki testowe dla tego podzadania. Twój wynik za to podzadanie będzie obliczony według następującej tabeli:

Warunek	Punkty
$40 \leq m \leq 60$	20
$26 \leq m \leq 39$	$25 + 1.5 \times (40 - m)$
$m = 25$	50
$m = 24$	55
$m = 23$	62
$m = 22$	70
$m = 21$	80
$m \leq 20$	90

Przykładowy program oceniający

Przykładowy program oceniający czyta dane wejściowe zgodnie z następującym formatem

- wiersz 1: N
- wiersz $2 + k$ ($0 \leq k$): $A[k]$ $B[k]$
- ostatni wiersz: -1

Każdy wiersz, z wyjątkiem pierwszego i ostatniego, określa pewien scenariusz. W wierszu $2 + k$ mamy scenariusz k. W scenariuszu k torba A zawiera $A[k]$ monet, a torba B zawiera $B[k]$ monet.

Przykładowy program oceniający wywołuje najpierw `devise_strategy(N)`. Wartość x to pomniejszona o jeden długość przekazanej tablicy. Następnie jeśli przykładowy program oceniający wykryje, że tablica przekazana przez `devise_strategy` nie spełnia założeń opisanych w Szczegółach impolementacyjnych, kończy działanie, wypisując jeden z następujących komunikatów:

- `s is an empty array`: s jest pustą tablicą (co nie odpowiada żadnej poprawnej strategii).
- `s[i] contains incorrect length`: Istnieje indeks i ($0 \leq i \leq x$) taki, że długość $s[i]$ jest różna od $N + 1$.
- `First element of s[i] is non-binary`: Istnieje indeks i ($0 \leq i \leq x$) taki, że $s[i][0]$ nie jest ani 0 ani 1.
- `s[i][j] contains incorrect value`: Istnieją indeksy i, j ($0 \leq i \leq x, 1 \leq j \leq N$) taki, że $s[i][j]$ jest spoza zakresu od -2 do x.

W przeciwnym razie przykładowy program oceniający generuje dwa wyjścia:

Najpierw wypisuje wynik Twojej strategii w następującym formacie:

- wiersz $1 + k$ ($0 \leq k$): wynik Twojej strategii dla scenariusza k. Jeśli zastosowanie strategii prowadzi do wyboru torby A jako mającej mniej monet, to wynikiem jest litera `A`. Jeśli zastosowanie strategii prowadzi do wyboru torby B jako mającej mniej monet, to wynikiem jest litera `B`. Jeśli zastosowanie strategii nie kończy się wyborem torby z mniejszą liczbą monet przez żadnego z więźniów, to wynikiem jest litera `X`.

Następnie przykładowy program oceniający zapisuje w bieżącym katalogu plik `log.txt` w następującym formacie

- wiersz $1 + k$ ($0 \leq k$): $w[k][0]$ $w[k][1]$ $\ldots$

Ciąg w wierszu $1 + k$ odpowiada scenariuszowi k i określa liczby zapisane na tablicy. Konkretnie, $w[k][l]$ jest liczbą zapisaną przez ($l - tego$ więźnia.