D. Садові прикраси

Назва задачі	Garden Decorations
Обмеження часу	7 с
Обмеження використання пам'яті	1024 МБ

Щодня, йдучи до школи і назад додому, Уляна проходить по вулиці з N будинками, пронумерованими від 0 до $N-1$. Наразі в будинку з номером i проживає житель з номером i. Для зміни обстановки жителі вирішили помінятися будинками один з одним. В будинок з номером i переїде житель a_i (який наразі живе в будинку з номером a_i).

У кожному будинку є статуетка птаха в саду, всі такі статуетки виглядають однаково. Статуетки можуть перебувати у двох можливих станах: з *відкритими* крилами (нібито птах летить) або з *закритими* крилами (нібито птах стоїть на землі). Жителі мають дуже сильні переваги щодо того, як повинні виглядати їхні статуетки птахів, і відмовляються переїжджати в новий будинок, поки статуетка в їхньому новому саду не буде виглядати так само, як у їхньому попередньому саду. Уляна хоче допомогти їм розмістити статуетки птахів так, щоб вони змогли переїхати.

Для цього вона робить наступне: коли вона йде по вулиці (чи то в школу, чи додому), вона спостерігає за птахами, яких проходить, і, можливо, коригує деякі з статуеток (відкриваючи чи закриваючи їхні крила). Оскільки її дні в школі та вдома дуже напружені, вона не пам'ятає стан птахів, яких бачила на своїх попередніх прогулянках. На щастя, вона записала список $a_0, a_1, ..., a_{N-1}$, тому знає, який житель куди переїжджає.

Допоможіть Уляні розробити стратегію, яка підкаже їй, стани яких птахів потрібно змінити, щоб налаштувати статуетки відповідно до вподобань жителів. Вона може пройти по вулиці не більше ніж 60 разів, але для досягнення кращого результату їй слід пройти по вулиці меншу кількість разів.

Імплементація

Це задача з багатьма запусками, що означає, що ваша програма буде виконуватись кілька разів.

При кожному запуску спочатку потрібно прочитати рядок з двома цілими числами w та N — номер прогулянки та кількість будинків відповідно. При першому запуску вашої програми

параметр w буде рівним 0, при другому параметр w буде рівним 1, і так далі (більше деталей пояснено нижче).

У другому рядку задано N цілих чисел $a_0, a_1, ..., a_{N-1}$, що означає, що особа, яка буде переїжджати в будинок з номером i, наразі живе в будинку з номером a_i. Числа a_i утворюють *перестановку*: тобто кожне число від 0 до $N-1$ зустрічається рівно один раз в масиві a. Зверніть увагу, що житель може не переїжджати; тобто можливим є $a_i = i$.

Жителі обмінюються будинками лише один раз. Це означає, що для фіксованого тесту значення N та масив a будуть однаковими для всіх запусків вашої програми.

Перший запуск.

При першому запуску вашої програми параметр w буде рівним 0. У цьому виконанні вам слід вивести одне ціле число W ($0 \le W \le 60$) — кількість разів, яку ви хочете, щоб Уляна пройшла повз будинки. Після цього ваша програма повинна завершитися. Далі ваша програма буде виконуватись ще W разів.

Наступні запуски.

У наступному запуску вашої програми $w = 1$; у наступному після нього $w = 2$; і так далі аж до останнього запуску, де $w = W$.

Після того, як ви прочитаєте w, N та $a_0, a_1, ..., a_{N-1}$, Уляна починає йти по вулиці.

- Якщо w непарне, Уляна йтиме від дому до школи, проходячи будинки у порядку $0, 1, ..., N-1$.

  Ваша програма тепер повинна прочитати рядок з одним цілим числом b_0, котре буде рівним або 0 (закрито), або 1 (відкрито) — поточний стан статуетки перед будинком з номером 0. Після того, як ви прочитаєте b_0, ви повинні вивести рядок з одним числом рівним або 0, або 1 — нове значення, яке ви хочете встановити для b_0.

  Потім ваша програма повинна прочитати рядок з одним цілим числом b_1 — станом статуетки перед будинком з номером 1; і вивести нове значення для b_1. Це продовжується для кожного з N будинків. Після того, як ви пройдете останній будинок (тобто ви прочитаєте та запишете b_{N-1}), ваша програма повинна завершитися.

  Зверніть увагу, що ваша програма зможе прочитати наступне значення b_{i+1} тільки після того як ви встановили значення b_i.

- Якщо w парне, Уляна йтиме від школи до дому — вона проходитиме будинки у порядку $N-1, N-2, ...0$.

Процес такий самий, як і при непарному w, за винятком того, що ви починаєте з читання та запису b_{N-1}, потім b_{N-2} і так далі до b_0.

Коли $w = 1$, вхідні значення $b_0, b_1, ..., b_{N-1}$ — це початковий стан статуеток птахів. Коли $w > 1$, вхідні значення $b_0, b_1, ..., b_{N-1}$ для вашої програми будуть такими, якими їх встановив попередній запуск вашої програми.

В кінці, після останнього запуску вашої програми, значення b_i повинно дорівнювати початковому значенню b_{a_i} для всіх i; якщо це буде не так, то ви отримаєте вердикт "Неправильна відповідь" (Wrong Answer).

Деталі

Якщо *сума* часу виконання $W + 1$ окремих запусків вашої програми перевищить обмеження часу, ваша відправка отримає вердикт "Перевищення ліміту часу" (Time Limit Exceeded).

Переконайтеся, що буфер стандартного виводу очищується після виводу кожного рядка, інакше ваша відправка може отримати вердикт "Перевищення ліміту часу" (Time Limit Exceeded). У Python це відбувається автоматично, якщо ви використовуєте `input()` для читання рядків. У C++, `cout << endl;` очищує буфер після друку рядка; якщо ви використовуєте printf, скористайтеся `fflush(stdout)`.

Обмеження та оцінювання

- $2 \leq N \leq 500$.
- Ви можете використати не більше ніж $W \leq 60$ запусків.

Ваше рішення буде протестоване на наборі груп, кожна з яких оцінюється в певну кількість балів. Кожна група містить певний набір тестів. Щоб отримати бали за певну групу, вам необхідно вирішити всі тести цієї групи.

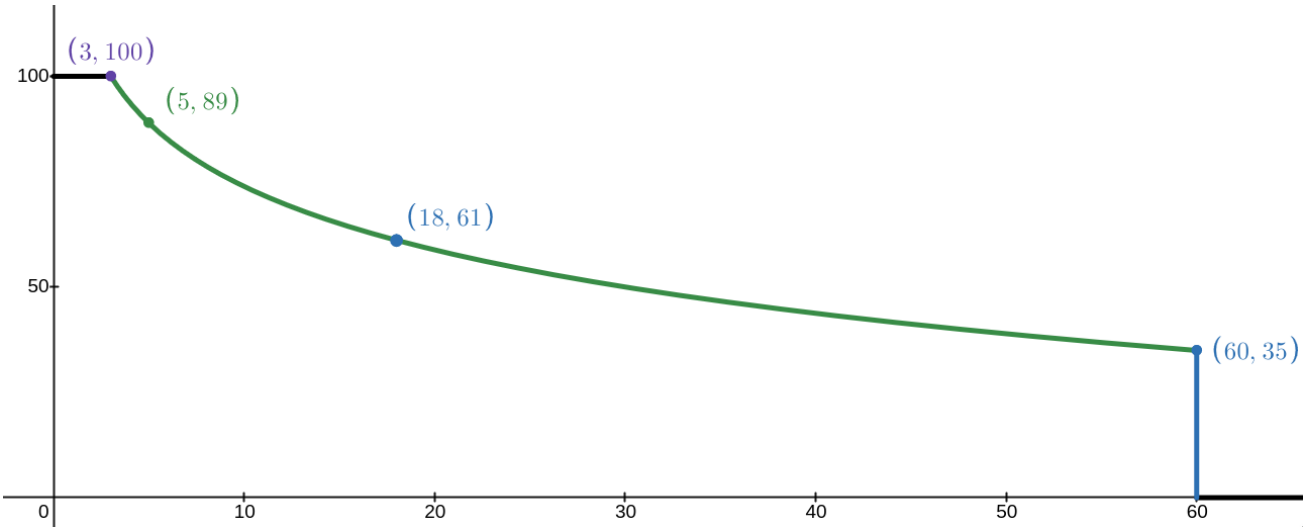
Група	Максимальний бал	Обмеження
1	10	$N = 2$
2	24	$N \leq 15$
3	9	$a_i = N - 1 - i$
4	13	$a_i = (i + 1) \bmod N$
5	13	$a_i = (i - 1) \bmod N$
6	31	Без додаткових обмежень

За кожну групу тестів, яку ваша програма вирішить правильно, ви отримаєте оцінку на основі наступної формули:

$$\text{score} = S_g \cdot \left(1 - \tfrac{1}{2}\log_{10}(\max(W_g, 3)/3)\right),$$

де S_g — максимальна оцінка для групи тестів, а W_g — максимальне значення W, використане для будь-якого тесту з групи. Ваша оцінка за кожну групу тестів буде округлена до найближчого цілого числа.

Графік нижче показує кількість балів, які ваша програма отримає в залежності від значення W, якщо вона вирішить всі групи тестів з однаковим значенням W. Зокрема, щоб отримати 100 балів за цю задачу, вам потрібно вирішити кожен тестовий випадок при $W \leq 3$.



Інструмент тестування

Щоб полегшити тестування вашого рішення, ми надаємо простий інструмент, який ви можете завантажити. Дивіться розділ “attachments” внизу сторінки задачі на Kattis. Використання цього інструменту не є обов'язковим. Зверніть увагу, що офіційна програма для оцінювання на Kattis відрізняється від тестувального інструменту.

Для використання інструменту створіть вхідний файл, наприклад, “sample1.in”, який має починатися з числа N, за яким слідує рядок з N числами, що задають перестановку, та інший рядок з N бітами (0 або 1), що вказують на початкові стани птахів. Наприклад:

```
6
1 2 0 4 3 5
1 1 0 0 1 0
```

Для програм на Python, наприклад, `solution.py` (зазвичай запускається як `pypy3 solution.py`):

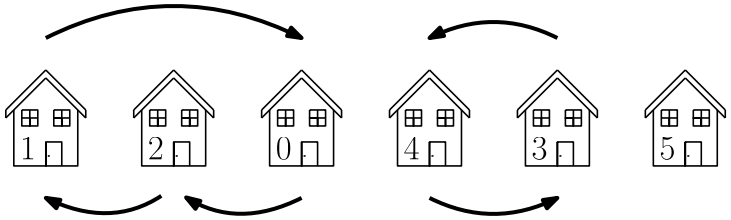
```
python3 testing_tool.py pypy3 solution.py < sample1.in
```

Для програм на C++, спочатку скомпілюйте її (наприклад, за допомогою `g++ -g -O2 -std=gnu++20 -static solution.cpp -o solution.out`) і потім запустіть:

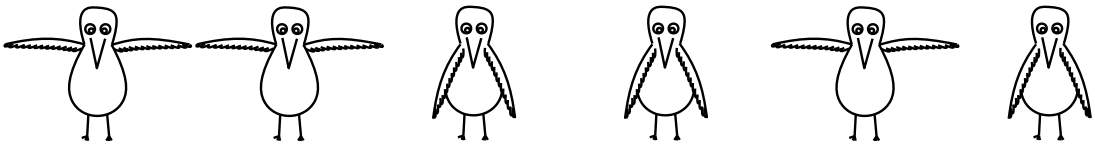
```
python3 testing_tool.py ./solution.out < sample1.in
```

Приклад

У прикладі нам задано наступну перестановку людей у будинках:

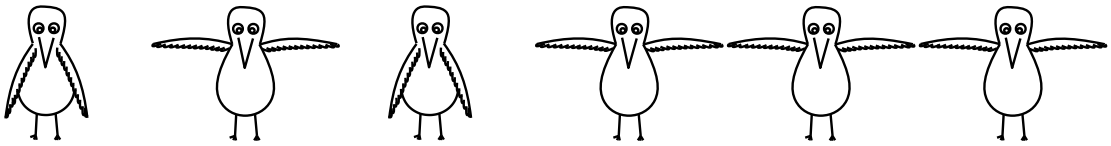


Перший раз, коли програма запускається (з $w = 0$), вона виводить $W = 2$, що означає, що Уляна пройде вулицею два рази (і відповідно програма буде запущена ще двічі). Перед першою прогулянкою стани птахів в садах виглядають наступним чином:



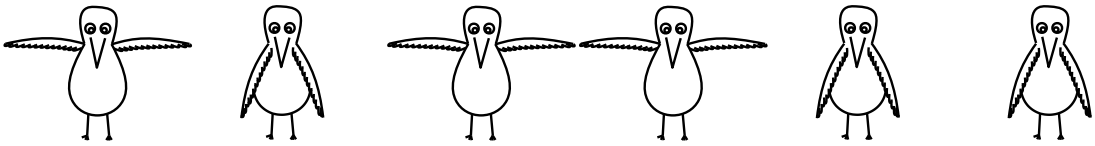
Потім програма запускається з $w = 1$: вказуючи на першу прогулянку Уляни. Вона проходить птахів один за одним, починаючи зліва, і, можливо, змінює їхні стани. Програма повинна вивести стан i-го птаха перед тим, як побачить стан $(i + 1)$-го птаха.

Після того, як Уляна прибуде до школи, стани птахів виглядатимуть так:



У фінальному запуску програми (з $w = 2$), Уляна йде додому зі школи. Пам'ятайте, що в цьому випадку вона пройде через птахів справа наліво і обробить їх у зворотному порядку! Це означає, що їй потрібно визначити стан i-го птаха перед тим, як побачити $(i - 1)$-го птаха.

Після того, як вона закінчить прогулянку, стани птахів виглядатимуть так:



Дійсно, це правильна конфігурація. Наприклад, птах на статуетці 3 (тобто четвертий зліва) відкритий (тепер $b_3 = 1$), що є правильним, оскільки людина 4 буде рухатися туди ($a_3 = 4$) і спочатку у неї була відкрита статуетка птаха (початково $b_4 = 1$).

Вивід градера	Ваш вивід
0 6	
1 2 0 4 3 5	
	2

Вивід градера	Ваш вивід
1 6	
1 2 0 4 3 5	
1	
	0
1	
	1
0	
	0
0	
	1
1	
	1
0	
	1

Вивід градера	Ваш вивід
2 6	
1 2 0 4 3 5	
1	
	0
1	
	0
1	
	1
0	
	1
1	
	0
0	
	1