D. Градинска декорация

Име	Garden Decorations
Time Limit	7 seconds
Memory Limit	1 gigabyte

Всеки ден, докато отива на училище и се връща у дома, Симона върви по улица с N къщи, номерирани с числата от 0 до $N-1$. В момента, къща с номер i е населена от човек с номер i. За разнообразие, жителите решават да разменят къщите помежду си. Човекът, който ще се премести в къща с номер i, е човекът с номер a_i (който в момента живее в къща с номер a_i).

Всяка къща има идентично изглеждаща статуя на птица в градината. Статуите имат две възможни положения - с *отворени* криле (символизираща птица в полет) или *затворени* (символизираща кацнала на земята птица). В момента, всяка птица е настроена в състоянието, в което собственика ѝ я предпочита. Жителите много държат на това как изглеждат техните статуи и отказват да се преместят в новата си къща, докато състоянието ѝ не стане според техните предпочитания.

Симона иска да помогне на жителите да променят състоянията на птиците, така че да могат да се преместят.

За тази цел тя прави следното: всеки път, когато преминава по улицата (на път за училище или обратно към вкъщи), тя наблюдава птиците, покрай които минава, една по една и евентуално коригира някои от статуите (чрез отваряне или затваряне на крилата им). Тъй като дните ѝ в училище и у дома са много натоварени, **тя не помни състоянието на птиците, които е видяла по време на предишните си разходки**. За щастие, тя е записала списъка $a_0, a_1, ..., a_{N-1}$, така че знае кой жител къде се премества.

Помогнете на Симона да измисли стратегия, която да ѝ каже кои птици да променя, за да настрои статуите според предпочитанията на жителите. Тя може да премине по улицата максимум 60 пъти, но за да постигне по-висок резултат, иска да премине по улицата по-малкък брой пъти.

Реализация

Това е *multirun* задача, което означава, че вашата програма ще бъде изпълнявана многократно.

При всяко изпълнение, първо трябва да прочетете от стандартния вход ред с две цели числа w и N, съответно индекса на разходката и броя на къщите. В първото изпълнение на вашата програма $w = 0$, във второто $w = 1$ и така нататък (за повече подробности вижте по-долу).

На втория ред са дадени N цели числа $a_0, a_1, ..., a_{N-1}$, което означава, че човекът, който ще се премести в къща i, в момента живее в къща a_i. Редицата a_i формира *пермутация*: тоест, всяко число от 0 до $N - 1$ се среща точно веднъж в редицата a_i. Обърнете внимание, че някои жители може да изберат да не се местят; т.е., позволено е $a_i = i$.

Жителите ще сменят къщите си само веднъж. Това означава, че за даден тестов случай стойността на N и редицата a_i ще бъдат същите за всички изпълнения на вашата програма.

Първо изпълнение

За първото изпълнение на вашата програма $w = 0$. В това изпълнение трябва просто да изведете едно цяло число W ($0 \leq W \leq 60$), броят пъти, които искате Симона да премине покрай къщите, след което да завърши.

След това вашата програма ще бъде изпълнявана W пъти.

Последващи изпълнения

В следващото изпълнение на вашата програма, $w = 1$; в следващото след това $w = 2$; и така нататък до последното изпълнение, когато $w = W$.

След като прочетете w, N и $a_0, a_1, ..., a_{N-1}$, Симона започва да ходи по улицата.

- Ако w е нечетно, Симона ще ходи от дома си до училище и ще премине покрай къщите в реда $0, 1, ..., N - 1$.

  Вашата програма трябва да прочете от стандартния вход ред с b_0, състоянието на статуята пред къща 0, която е или 0 (затворена) или 1 (отворена). След като прочетете b_0, трябва да изведете ред или с 0, или с 1, новата стойност, която искате да зададете на b_0.

  След това вашата програма трябва да прочете ред с b_1, състоянието на статуята пред къща 1; и да изведе новата стойност на b_1. Това продължава за всяка от N-те къщи. След като Симона премине последната къща (т.е., прочетете и напишете b_{N-1}) вашата програма трябва да завърши.

Обърнете внимание, че вашата програма може да прочете следващата стойност b_{i+1} само след като е извела стойността на b_i.

- Ако w е четно, Симона се връща от училище към дома си и преминава покрай къщите в обратен ред, т.е $N-1, N-2, ..., 0$.

  Процесът е същият като при нечетно w, с изключение на това, че започвате с прочитане и записване на b_{N-1}, след това b_{N-2} и така нататък до b_0.

Когато $w = 1$, входните стойности $b_0, b_1, ..., b_{N-1}$ са оригиналното състояние на статуите на птиците. Когато $w > 1$, входните стойности $b_0, b_1, ..., b_{N-1}$ на вашата програма ще бъдат тези, които предишното изпълнение на вашата програма е задало.

В края, след последното изпълнение на вашата програма, стойността на b_i трябва да бъде равна на оригиналната стойност на b_{a_i} за всички i, в противен случай ще получите грешен отговор.

Заблежка

Ако *сумата* от времето на изпълнение на $W+1$ отделни изпълнения на вашата програма надвиши времевия лимит, вашето решение ще бъде оценено с превишено време за изпълнение.

Уверете се, че `flush`-вате стандартния изход след всяка линия, в противен случай вашата програма може да бъде оценена с превишено време за изпълнение. В Python това се случва автоматично, стига да използвате `input()` за четене на редове. В C++, `cout << endl;` извършва допълнително изпразване на буфера освен извеждането на нов ред; ако използвате `printf`, използвайте `fflush(stdout)`.

Ограничения и Оценяване

- $2 \leq N \leq 500$.
- Може да използвате най-много 60 кръга ($W \leq 60$).

Вашето решение ще бъде тествано с набор от тестови групи, всяка за определен брой точки. Всяка тестова група съдържа набор от тестови случаи. За да получите точки за тестовата група, трябва да решите всички тестови случаи в тестовата група.

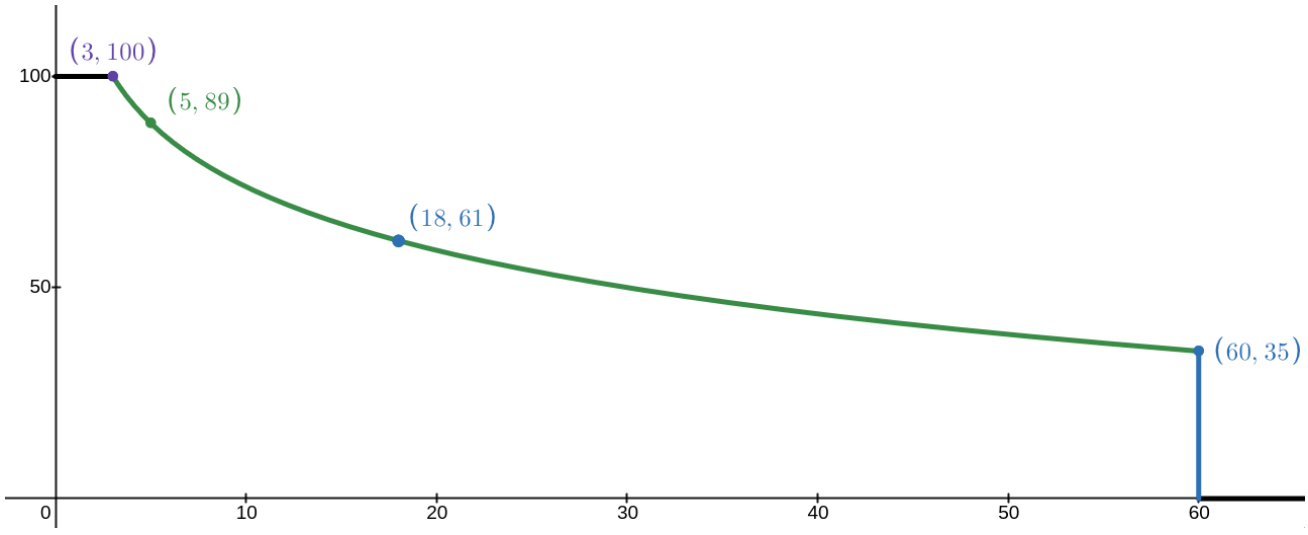
Група	Максимален резултат	Ограничения
1	10	$N = 2$
2	24	$N \leq 15$
3	9	$a_i = N - 1 - i$
4	13	$a_i = (i + 1) \bmod N$
5	13	$a_i = (i - 1) \bmod N$
6	31	Без допълнителни ограничения

За всяка тестова група, която вашата програма решава правилно, ще получите точките въз основа на следната формула:

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

където S_g е максималният резултат за тестовата група, а W_g е максималната стойност на W, използвана за всеки тестов случай в тестовата група. Вашият резултат за всяка тестова група ще бъде закръглен до най-близкото цяло число.

Графиката по-долу показва броя на точките, в зависимост от W, които вашата програма ще получи, ако реши всички тестови случаи с една и съща стойност на W. По-специално, за да постигнете резултат от 100 точки по тази задача, трябва да решите всеки тестов случай с $W \leq 3$.



Инструмент за тестване

За да улесните тестването на вашето решение, предоставяме прост инструмент, който можете да изтеглите. Вижте “attachments” в долната част на страницата на задачата в Kattis.

Не е задължително да използвате инструмента. Обърнете внимание, че официалната програма за оценяване на Kattis е различна от инструмента за тестване.

За да използвате инструмента, създайте входен файл, например “sample1.in”, който трябва да започва с число N, последвано от ред с N числа, определящи пермутацията, и трети ред с N бита (0 или 1), задаващи началните състояния на птиците. Например:

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

За Python програми, изпълнете `solution.py` (обикновено се изпълнява като `pypy3 solution.py`):

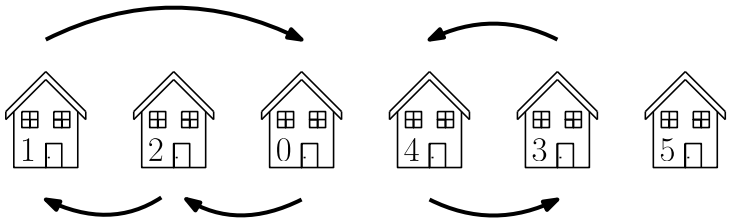
```
python3 testing_tool.py pypy3 solution.py < sample1.in
```

За C++ програми, първо компилирайте (например с `g++ -g -O2 -std=gnu++20 -static solution.cpp -o solution.out`) и след това изпълнете:

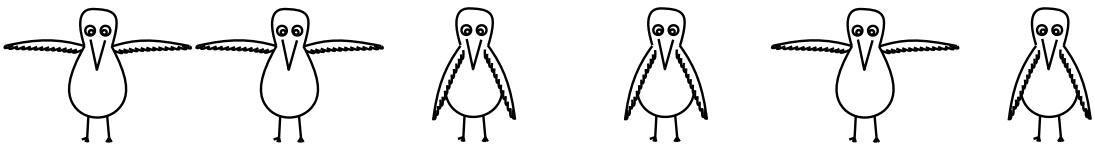
```
python3 testing_tool.py ./solution.out < sample1.in
```

Пример

В примера ни е дадена следната пермутация на хората в къщите:

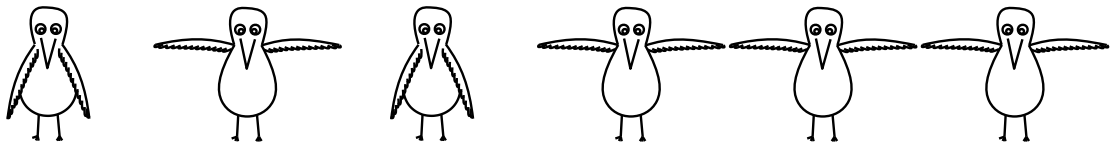


Първият път, когато програмата се изпълни (с $w = 0$), тя извежда $W = 2$, което означава, че Симона ще премине по улицата два пъти (и програмата ще бъде изпълнена още два пъти). Преди първото преминаване, птиците в градините изглеждат по следния начин:



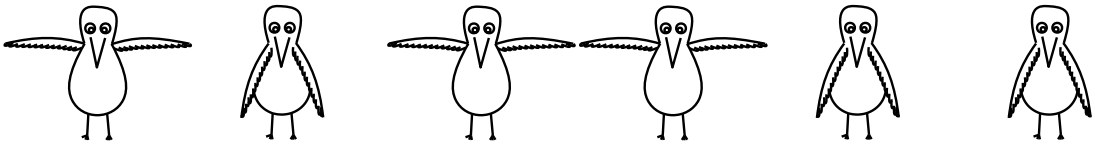
След това програмата се изпълнява с $w = 1$: което указва първото преминаване на Симона. Тя преминава през птиците една по една, започвайки отляво, и евентуално променя тяхното състояние. Програмата трябва да изведе състоянието на i-та птица преди да видим $(i + 1)$-та птица.

След като Симона пристигне в училище, състоянието на птиците изглежда така:



В последното изпълнение на програмата (с $w = 2$), Симона се връща у дома от училище. Помнете, че в този случай тя ще премине през птиците отдясно наляво и ще обработи тях в обратен ред! Това означава, че тя трябва да определи състоянието на i-та птица преди да види $(i - 1)$-та птица.

След като завърши своето преминаване, птиците изглеждат така:



Наистина, това е правилната конфигурация. Например, статуята на птица 3 (т.е. четвъртата отляво) е отворена (сега $b_3 = 1$), което е правилно, тъй като човек 4 ще се премести там ($a_3 = 4$) и първоначално е имала отворена статуя на птица (първоначално $b_4 = 1$).

изход на грейдъра	вашият изход
0 6	
1 2 0 4 3 5	
	2

изход на грейдъра	вашият изход
1 6	
1 2 0 4 3 5	
1	
	0
1	
	1
0	
	0
0	
	1
1	
	1
0	
	1

изход на грейдъра	вашият изход
2 6	
1 2 0 4 3 5	
1	
	0
1	
	0
1	
	1
0	
	1
1	
	0
0	
	1