D. 庭の飾り付け (Garden Decorations)

問題名	庭の飾りつけ (Garden Decorations)
実行時間制限	7 秒
メモリ制限	1 GB

Detje は，登校と下校の際に毎日，0 から $N-1$ までの番号が付けられた N 軒の家が一列に並んでいる街道を歩く．現在，家 i には住民 i が住んでいる．さて，景観の変化のため，人々は互いに転居することを決めた．具体的には，住民 a_i (現在家 a_i に住んでいる住民) が家 i に転居することを決めた．

ここで，各家の庭には鳥の像が設置されている．鳥の像には 2 つの状態があり，これらは (鳥が飛んでいるように) 羽が *開いている状態* と，(鳥が地面に立っているように) 羽が *閉じている状態* である．各住民は，鳥の像がどう見えるべきかについて極めて強い好みを持っている．住民 i が好きな鳥の状態は，現時点で家 i の庭にある鳥の状態である．住民たちは，全住民における転居後の家の庭にある鳥の状態が好きな状態になるまで，転居を断り続ける．そこで Detje は，鳥の像の状態を変更することで，転居を手伝うことにした．

転居を手伝うため，彼女は次のことを行う．彼女が街道を歩くとき (登校および下校の際)，彼女は通過した鳥の像を順番に観察し，いくつかの鳥の像の状態を (羽を開いたり閉じたりすることによって) 随時変更する．しかし，彼女は学校と自宅で忙しい日々を過ごしているため，**彼女は前回歩いたときの鳥の状態を覚えていない．**一方，幸運なことに，彼女は整数 $a_0, a_1, ..., a_{N-1}$ を紙に書いており，どの住民がどの家に転居するのかを知っている．

各住民の好みに合うような像の変更を行うために，どの像の状態を操作するかを伝える戦略を実装して Detje を手伝え．なお，彼女は街道を最大 60 回歩くことができるが，高い得点を得るためには，より少ない回数にする必要がある．

実装

この問題は複数回実行の問題である．すなわち，あなたのプログラムは複数回実行される．

各実行では，あなたは最初に 2 つの整数 w と N を入力しなければならない．ここで，w は何回目の歩行かを指す整数であり，N は家の数である．初回の実行では $w = 0$ であり，2 回目の実行では $w = 1$ であり，それ以降も同様である (詳細は後述)．

入力の 2 行目には，N 個の整数 $a_0, a_1, ..., a_{N-1}$ が書かれている．これは，現在家 a_i に住んでいる住民が家 i に転居することを意味する．ここで a_i は順列であり，すなわち 0 から $N-1$ までの整数が 1 回ずつ出現する．なお，各住民は転居しないという選択をすることも可能であり，すなわち $a_i = i$ となることもある．

各住民は一度しか転居を行わない．すなわち，1 つのテストケースにおいて，N および a_i の値はすべての実行で同じである．

初回の実行

初回の実行では $w = 0$ となる．この実行では，あなたのプログラムは単に， Detje が歩くべき回数を表す整数 W ($0 \leq W \leq 60$) を出力しなければならない．そして出力の後，あなたのプログラムは実行を終了しなければならない．その後，あなたのプログラムは追加で W 回実行される．

それ以降の実行

次の実行では $w = 1$，その次の実行では $w = 2$ となる．以降も同様であり，最後の実行では $w = W$ となる．

あなたのプログラムが整数 $w, N, a_0, a_1, ..., a_{N-1}$ を入力した後，Detje は街道の歩行を始める．

- もし w が奇数である場合，Detje は自宅を出発し，学校に向けて歩く．彼女は家 $0, 1, ..., N-1$ の順に通過する．

  まず，あなたのプログラムは b_0 を入力しなければならない．ここで，b_0 は家 0 の庭の像の状態を表す整数であり，0 (閉じている状態) または 1 (開いている状態) である．そして整数 b_0 を読み込んだ後，あなたは変更後の b_0 の値を出力することで，像の状態を変更するかどうかを決めなければならない．

  次に，あなたのプログラムは b_1 を入力しなければならない．ここで，b_1 は家 1 の庭の像の状態を表す整数である．その後，変更後の b_1 の値を出力しなければならない．同様の手順は家 $2, 3, ..., N-1$ についても続く．Detje が最後の家を通過した後 (すなわち，b_{N-1} に関する入力と出力を行った後)，**あなたのプログラムは終了しなければならない．**

  なお，あなたのプログラムは変更後の b_i の値を出力するまで，b_{i+1} の値を入力することができない．

- もし w が偶数である場合，Detje は学校を出発し，自宅に向けて歩く．彼女は家 $N-1, N-2, ..., 0$ の順に通過する．すなわち，あなたのプログラムは $b_{N-1}, b_{N-2}, ..., b_0$ の順に入出力を行う．

$w = 1$ を満たす実行では，入力される値 $b_0, b_1, ..., b_{N-1}$ は「鳥の像の最初の状態」である (各住民の好きな状態でもある)．$w > 1$ を満たす実行では，入力される値 $b_0, b_1, ..., b_{N-1}$ は「前回の実行で出力された値」である．

最終的に，プログラムの最後の実行が終わった時点で，すべての i について b_i の値が b_{a_i} の初期値と等しくなければならない．さもなくば，Wrong Answer と判定される．

詳細

もし $W+1$ 回の実行における実行時間の *総和* が実行時間制限を超過した場合，Time Limit Exceeded と判定される．

各行の出力の最後には，必ず標準出力を flush せよ．さもなくば，Time Limit Exceeded と判定される．Python の場合，`input()` によって自動的に flush される．C++ の場合，`cout << endl;` によって，改行されるとともに自動的に flush される．もし printf を使用する場合，`fflush(stdout)` を用いよ．

制約・採点形式

- $2 \leq N \leq 500$.
- $W \leq 60$ 回の歩行しか行うことができない．

あなたの解答は各小課題ごとに評価され，小課題にはそれぞれ配点が割り当てられている．各小課題は複数のテストケースからなる．各小課題について得点を得るためには，その小課題に含まれるすべてのテストケースに正解する必要がある．

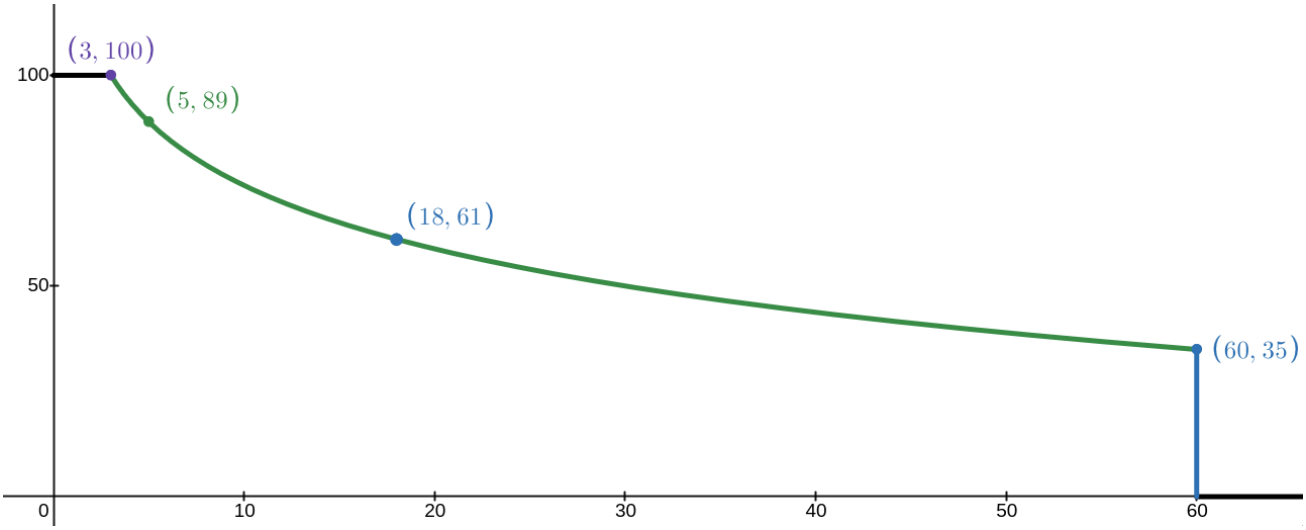
小課題	配点	制約
1	10	$N = 2$
2	24	$N \leq 15$
3	9	$a_i = N - 1 - i$
4	13	$a_i = (i + 1) \bmod N$
5	13	$a_i = (i - 1) \bmod N$
6	31	追加の制約はない

正解した各小課題について，次式に基づく得点が与えられる．

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

ここで，S_g は各小課題の配点であり，W_g はこの小課題を満たすテストケースで指定された W の最大値を表す．各小課題におけるあなたの得点は，最も近い整数に四捨五入される．

以下のグラフは，すべての小課題について同じ整数 W を指定した場合における，整数 W と得点の関係を示している．特に，この問題で 100 点を得るためには，すべてのテストケースについて $W \leq 3$ で解く必要がある．



テストのためのツール

あなたが解法をテストすることを容易にするため，簡単なツールがダウンロードできるようになっている．Kattis 問題ページの下部の "attachments" の項を見よ．ここで，本ツールを必ずしも使う必要はない．なお，Kattis で使用される実際の採点プログラムは，本ツールとは異なる．

本ツールを使うには，まず "sample1.in" のような入力ファイルを作成しなければならない．この入力ファイルでは，最初の行に整数 N，次の行に順列を表す N 個の整数，その次の行に鳥の像の初期状態 (0 または 1) を N ビット指定しなければならない．具体例を以下に示す:

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

Python の場合，例えば `solution.py` の場合，(通常は `pypy3 solution.py` とすれば実行されるが) 以下を実行しなければならない．

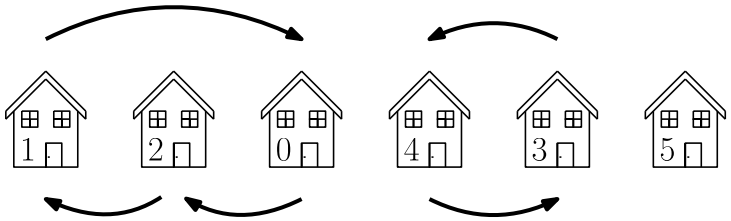
```
python3 testing_tool.py pypy3 solution.py < sample1.in
```

C++ の場合，まずコンパイルを行い (例: `g++ -g -O2 -std=gnu++20 -static solution.cpp -o solution.out`)，その後以下を実行しなければならない．

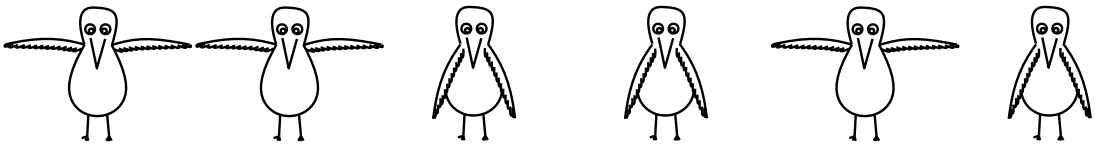
```
python3 testing_tool.py ./solution.out < sample1.in
```

例

サンプルテストケースでは，家の住民に関する順列として以下が入力される．

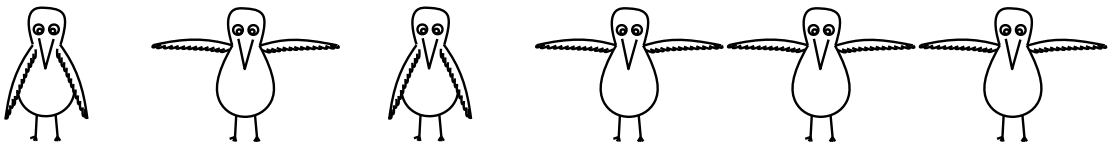


サンプルプログラムの初回の実行 ($w = 0$) では，$W = 2$ が出力される．これは，Detje が 2 回の歩行を行う (すなわち，プログラムは追加で 2 回実行される) ことを意味する．最初の歩行の前の時点において，庭の鳥の状態は下図のようになっている．



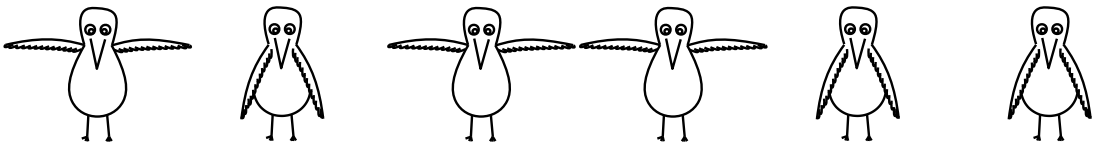
その後，プログラムは $w = 1$ で実行される (Detje の 1 回目の歩行を意味する)．彼女は鳥の像を左から順に通過し，状態を随時変更する．サンプルプログラムは，家 $i + 1$ の鳥に関する情報を入力する前に，家 i の鳥の変更後の状態を出力しなければならない．

Detje が学校に到着した後の時点で，鳥の状態は下図のようになっている．



プログラムの最後の実行 ($w = 2$) では，Detje は学校から出発して家に戻る．この実行では，彼女は鳥の像を右から順に通過し，つまり逆順に処理が行われる！　すなわち，彼女は家 $i - 1$ の鳥の状態を見る前に，家 i の鳥の変更後の状態を決めなければならない．

彼女が最後の歩行を終えた後の時点で，鳥の状態は下図のようになっている．



これは正しい状態になっている．たとえば，家 3 の鳥の像 (左から 4 番目) は開いている状態であるが (現在 $b_3 = 1$ である)，住民 4 がこの家に転居する予定であり ($a_3 = 4$)，この家の庭には最初の時点において，開いた状態の鳥の像がある (最初 $b_4 = 1$ である)．

採点プログラムの出力	あなたのプログラムの出力
0 6	
1 2 0 4 3 5	
	2

採点プログラムの出力	あなたのプログラムの出力
1 6	
1 2 0 4 3 5	
1	
	0
1	
	1
0	
	0
0	
	1
1	
	1
0	
	1

採点プログラムの出力	あなたのプログラムの出力
2 6	
1 2 0 4 3 5	
1	
	0
1	
	0
1	
	1
0	
	1
1	
	0
0	
	1