



Mechanische Puppe

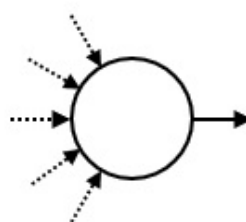
Eine mechanische Puppe ist eine Figur, die automatisch eine ihr eigene Abfolge von Bewegung ausführt. Seit Jahrhunderten werden in Japan mechanische Puppen gebaut.

Die Bewegungen einer mechanischen Puppe werden von einem **Schaltkreis** gesteuert, der aus **Schaltelementen** besteht. Die Schaltelemente sind untereinander mit Röhren verbunden. Jedes Schaltelement besitzt einen oder zwei **Ausgänge** und beliebig viele **Eingänge** (möglicherweise null). Jede Röhre verbindet einen Ausgang eines Elements mit dem Eingang desselben oder eines anderen Elements. Jeder Ein- und Ausgang ist mit je genau einer Röhre verbunden.

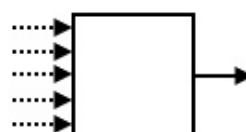
Um die Bewegungen der Puppe zu beschreiben, stellen wir uns einen **Ball** vor, der in eines der Schaltelemente gelegt wird. Der Ball bewegt sich durch den Schaltkreis. In jedem Schritt verlässt der Ball ein Element durch eines seiner Ausgänge, folgt der mit dem entsprechenden Ausgang verbundenen Röhre und tritt in das Schaltelement am anderen Ende der Röhre ein.

Es gibt drei Arten von Schaltelementen: **Ursprung**, **Auslöser** und **Weiche**. Es gibt genau einen Ursprung, M Auslöser und S Weichen (S darf 0 sein). Der Wert von S ist von dir festzulegen. Jedes Element hat eine eindeutige Seriennummer.

Der Ursprung ist das Schaltelement, in welchem sich der Ball zu Beginn befindet. Es hat genau einen Ausgang. Seine Seriennummer lautet 0.

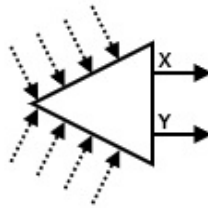


Ein Auslöser lässt die Puppe bei Eintritt des Balls eine bestimmte Bewegung ausführen. Jeder Auslöser hat genau einen Ausgang. Die Auslöser haben die Seriennummern von 1 bis M .



Jede Weiche hat zwei Ausgänge, die 'X' bzw. 'Y' genannt werden. Der **Zustand** einer

Weiche ist entweder 'X' oder 'Y'. Nachdem ein Ball eine Weiche betreten hat, verlässt er die Weiche über den Ausgang, der ihrem Zustand entspricht. Danach verändert sich der Zustand der Weiche zum jeweils anderen. Zu Beginn ist jede Weiche im Zustand 'X'. Weichen haben die Seriennummern von -1 bis $-S$.



Du erhältst die Anzahl M der Auslöser. Weiterhin erhältst Du eine Folge A der Länge N , die aus Seriennummern von Auslösern besteht. Jeder Auslöser kann nullmal, einmal oder mehrmals in der Folge A vorkommen. Deine Aufgabe ist es, einen Schaltkreis zu entwerfen, der den folgenden Bedingungen genügt:

- Der Ball kehrt irgendwann zum Ursprung zurück.
- Wenn der Ball das erste Mal zum Ursprung zurückkehrt, ist jede Weiche im Zustand 'X'.
- Der Ball kehrt das erste Mal zum Ursprung zurück, nachdem insgesamt genau N Auslöser durchlaufen wurden. Die Seriennummern der Auslöser in der Reihenfolge des Durchlaufens lauten genau A_0, A_1, \dots, A_{N-1} .
- Sei P die Gesamtzahl aller Zustandsänderungen von Weichen, die von dem Ball verursacht werden, bevor er das erste Mal zum Ursprung zurückkehrt. Dann ist P nicht größer als 20 000 000.

Gleichzeitig möchtest Du nicht zu viele Weichen benutzen.

Implementierungshinweise

Implementiere die folgende Funktion:

```
create_circuit(int M, int[] A)
```

- M : die Anzahl der Auslöser.
- A : ein Array der Länge N mit der Folge der Seriennummern der Auslöser in der Reihenfolge, in welcher der Ball diese durchlaufen soll.
- Diese Funktion wird genau einmal aufgerufen.
- Beachte, dass N die Länge des Arrays A ist und entsprechend den allgemeinen Implementierungshinweisen erhalten werden kann.

Dein Programm soll die folgende Funktion aufrufen, um die Lösung zu übermitteln:

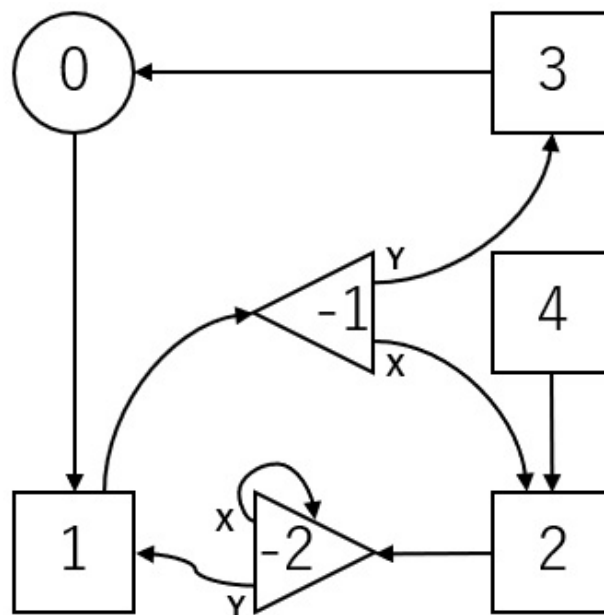
```
answer(int[] C, int[] X, int[] Y)
```

- C : ein Array der Länge $M + 1$. Der Ausgang von Element i ($0 \leq i \leq M$) ist mit Element $C[i]$ verbunden.
- X, Y : Arrays der Länge S , wobei S die Anzahl der Weichen ist. Der Ausgang 'X' von Weiche $-j$ ($1 \leq j \leq S$) ist mit Schaltelement $X[j - 1]$ verbunden und der Ausgang 'Y' mit Schaltelement $Y[j - 1]$.
- Jedes Element von C, X und Y muss eine ganze Zahl zwischen $-S$ und M einschließlich sein.
- S muss kleiner oder gleich 400 000 sein.
- Diese Funktion muss genau einmal aufgerufen werden.
- Der durch C, X und Y definierte Schaltkreis muss den obigen Bedingungen der Aufgabenstellung genügen.

Sollte mindestens eine dieser Bedingungen verletzt sein, wird Dein Programm mit **Wrong Answer** bewertet. Andernfalls wird Dein Programm mit **Accepted** bewertet und Deine Punktzahl ergibt sich aus S (siehe Teilaufgaben).

Beispiel

Seien $M = 4, N = 4$ und $A = [1, 2, 1, 3]$. Der Grader ruft `create_circuit(4, [1, 2, 1, 3])` auf.



Das obige Bild zeigt einen Schaltkreis, welcher dem Aufruf `answer([1, -1, -2, 0, 2], [2, -2], [3, 1])` entspricht. Die Zahlen im Bild entsprechen den Seriennummern der Schaltelemente.

Es werden zwei Weichen benutzt. Somit ist $S = 2$.

Zu Beginn sind die beiden Weichen -1 und -2 im Zustand 'X'. Der Ball bewegt sich wie folgt:

$0 \longrightarrow 1 \longrightarrow -1 \xrightarrow{X} 2 \longrightarrow -2 \xrightarrow{X} -2 \xrightarrow{Y} 1 \longrightarrow -1 \xrightarrow{Y} 3 \longrightarrow 0$

- Beim ersten Betreten von Weiche -1 findet der Ball die Weiche im Zustand 'X' vor. Daher bewegt er sich in Richtung von Schalter 2. Dann ändert sich der Zustand von Weiche -1 zu 'Y'.
- Beim zweiten Betreten von Weiche -1 findet der Ball die Weiche im Zustand 'Y' vor. Daher bewegt er sich in Richtung von Schalter 3. Dann ändert sich der Zustand von Weiche -1 zu 'X'.

Wenn der Ball zum ersten Mal zum Ursprung zurückkehrt, hat er die Auslöser 1, 2, 1, 3 durchlaufen und beide Weichen sind im Zustand 'X'. P hat den Wert 4. Folglich genügt der Schaltkreis den Anforderungen.

Die Datei `sample-01-in.txt` im ZIP-Archiv unter *Attachments* entspricht diesem Beispiel. Dort findest Du auch weitere Beispieleingaben.

Einschränkungen

- $1 \leq M \leq 100\,000$
- $1 \leq N \leq 200\,000$
- $1 \leq A_k \leq M$ ($0 \leq k \leq N - 1$)

Teilaufgaben

Die Einschränkungen und Punktzahlen für die einzelnen Teilaufgaben sind wie folgt:

1. (2 Punkte) Für jedes i ($1 \leq i \leq M$), kommt die Zahl i höchstens einmal in der Folge A_0, A_1, \dots, A_{N-1} vor.
2. (4 Punkte) Für jedes i ($1 \leq i \leq M$), kommt die Zahl i höchstens zweimal in der Folge A_0, A_1, \dots, A_{N-1} vor.
3. (10 Punkte) Für jedes i ($1 \leq i \leq M$), kommt die Zahl i höchstens viermal in der Folge A_0, A_1, \dots, A_{N-1} vor.
4. (10 Punkte) $N = 16$
5. (18 Punkte) $M = 1$
6. (56 Punkte) Keine weiteren Einschränkungen.

Wird Dein Programm für einen Testfall mit **Accepted** bewertet, so ergibt sich Deine Punktzahl aus dem Wert von S wie folgt:

- Ist $S \leq N + \log_2 N$, so erhältst Du volle Punktzahl für diesen Testfall.
- Für jeden Testfall in Teilaufgaben 5 und 6 erhältst Du Teilpunkte, falls $N + \log_2 N < S \leq 2N$ ist. Die Punktzahl für den Testfall ergibt sich als $0.5 + 0.4 \times \left(\frac{2N - S}{N - \log_2 N} \right)^2$ multipliziert mit der Punktzahl der Teilaufgabe.
- Sonst ist die Punktzahl 0.

Die Punktzahl einer Teilaufgabe ist das Minimum der Punktzahlen ihrer Testfälle.

Beispielgrader

Der Beispielgrader liest die Eingabe in folgendem Format ein:

- Zeile 1: $M N$
- Zeile 2: $A_0 A_1 \dots A_{N-1}$

Der Beispielgrader erstellt drei Ausgaben. Erstens gibt der Beispielgrader Deine Antwort in folgendem Format in die Datei `out.txt` aus.

- Zeile 1: S
- Zeile $2 + i$ ($0 \leq i \leq M$): $C[i]$
- Zeile $2 + M + j$ ($1 \leq j \leq S$): $X[j - 1] Y[j - 1]$

Zweitens simuliert der Beispielgrader die Bewegungen des Balles. Er gibt die Seriennummern der Schaltelemente, die der Ball durchlaufen hat, in der entsprechenden Reihenfolge in die Datei `log.txt` aus.

Drittens gibt der Beispielgrader die Bewertung Deiner Antwort in die Standardausgabe aus.

- Falls Dein Programm mit **Accepted** bewertet wird, gibt der Beispielgrader S und P im Format `Accepted: S P` aus.
- Falls Dein Programm mit **Wrong Answer** bewertet wird, gibt der Beispielgrader `Wrong Answer: MSG` aus, mit folgender Bedeutung:
 - `answered not exactly once`: Die Funktion `answer` wurde nicht exakt einmal aufgerufen.
 - `wrong array length`: Die Länge von C ist nicht $M + 1$ oder die Längen von X und Y sind verschieden.
 - `over 400000 switches`: S ist größer als 400 000.
 - `wrong serial number`: Ein Element von C , X , oder Y ist kleiner als $-S$ oder größer als M .
 - `over 20000000 inversions`: Der Ball kommt nicht innerhalb von 20 000 000 Zustandswechseln von Weichen zum Ursprung zurück.
 - `state 'Y'`: Eine Weiche ist im Zustand 'Y', wenn der Ball erstmalig zum Ausgangspunkt zurückkehrt.
 - `wrong motion`: Die Auslöser der Bewegungen unterscheiden sich von der Sequenz A .

Beachte, dass der Beispielgrader möglicherweise keine `out.txt` und/oder `log.txt` Datei erstellt, wenn Dein Programm mit `Wrong Answer` bewertet wird.