# Boneka Mekanik

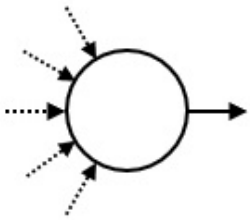
Boneka Mekanik adalah sebuah boneka yang secara otomatis mengulang sebuah urutan gerakan. Di Jepang, banyak boneka mekanik telah dibuat sejak zaman kuno.

Gerakan dari boneka mekanik dikontrol oleh sebuah **sirkuit** yang terdiri dari banyak **komponen**. Komponen-komponen tersebut terhubung dengan pipa. Setiap komponen memiliki satu atau dua **keluaran** dan dapat memiliki berapapun (bisa saja tidak ada) **masukan**. Setiap pipa menghubungkan sebuah keluaran ke sebuah masukan pada komponen yang sama atau komponen yang berbeda. Tepat sebuah pipa terhubung dengan setiap masukan dan setiap keluaran.

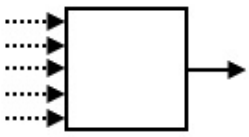
Untuk menjelaskan bagaimana boneka tersebut bekerja, bayangkan sebuah **bola** yang ditempatkan pada salah satu komponen. Bola tersebut akan menelusuri sirkuit tersebut. Pada setiap komponen, bola tersebut akan keluar dari salah satu keluaran, menelusuri pipa yang terhubung dengan keluaran tersebut dan masuk ke komponen yang ada di ujung yang lain. Bola tersebut tidak akan berhenti.

Terdapat tiga jenis komponen: **origin**, **trigger**, dan **switch**. Terdapat tepat sebuah *origin*, $M$ *trigger*, dan $S$ *switch* ($S$ bisa jadi 0). Anda harus menentukan nilai $S$. Setiap komponen memiliki nomor seri yang unik.

*Origin* adalah komponen dimana bola tersebut diletakkan pertama kali. Komponen tersebut memiliki sebuah keluaran dan nomor serinya adalah 0.

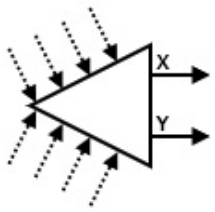


Sebuah *trigger* menyebabkan boneka tersebut melakukan sebuah gerakan spesifik ketika bola memasuki komponen tersebut. Setiap *trigger* memiliki sebuah keluaran. *Trigger* memiliki nomor seri dari 1 hingga $M$.



Setiap *switch* memiliki dua buah keluaran, 'X' dan 'Y'. **Status** dari sebuah *switch*

adalah 'X' atau 'Y'. Setelah bola tersebut memasuki sebuah *switch*, bola tersebut akan keluar dari keluaran yang ditentukan dari *status switch* tersebut. Setelah itu, *status* dari *switch* tersebut akan berubah ke *status* yang lain. Awalnya, *status* dari semua *switch* adalah 'X'. Nomor seri dari *switch* dimulai dari $-1$ hingga $-S$.




Anda diberikan $M$ buah *trigger*. Anda juga diberikan sebuah urutan $A$ dengan panjang $N$ yang setiap elemennya merupakan sebuah nomor seri sebuah *trigger*. Setiap *trigger* bisa jadi muncul beberapa (mungkin 0) kali di $A$. Tugas Anda adalah membuat sebuah sirkuit yang memenuhi kondisi berikut:

- Bola tersebut kembali ke *origin* setelah beberapa langkah.
- Ketika bola tersebut kembali ke *origin*, *status* semua *switch* adalah 'X'.
- Bola tersebut kembali ke *origin* untuk pertama kalinya setelah memasuki tepat $N$ buah *trigger*. Urutan nomor seri *trigger* yang dimasuki adalah $A_0, A_1, \ldots, A_{N-1}$.
- Misalkan $P$ adalah total perubahan *status* dari semua *switch* yang diakibatkan oleh bola tersebut sebelum kembali ke origin untuk pertama kalinya. Nilai dari $P$ tidak lebih dari $20\,000\,000$.

Pada waktu yang sama, Anda tidak ingin menggunakan terlalu banyak *switch*.

## Detail implementasi

Anda harus mengimplementasikan prosedur berikut:

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

- `M`: banyaknya *trigger*.
- `A`: array dengan panjang $N$, merepresentasikan urutan nomor seri *trigger* yang perlu dimasuki.
- Prosedur ini dipanggil tepat sekali.
- Perhatikan bahwa nilai dari $N$ adalah panjang dari array $A$ dan bisa didapatkan seperti yang dicantumkan di pemberitahuan implementasi.

Program Anda harus memanggil prosedur berikut untuk memberikan jawaban:

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

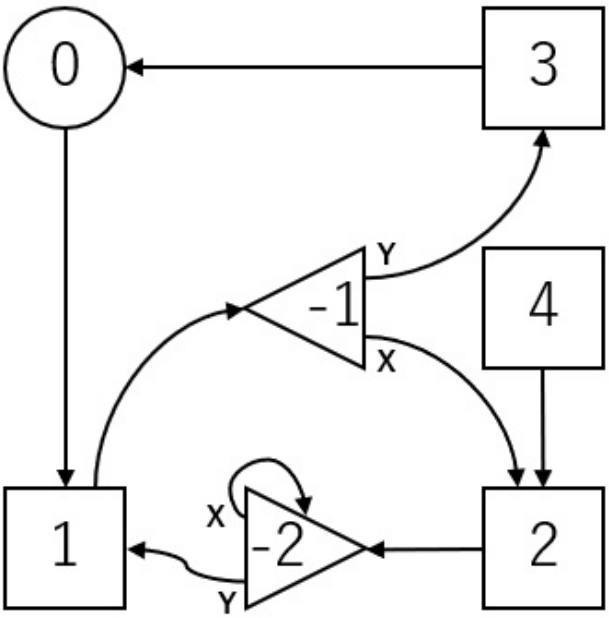
- `C`: array dengan panjang $M + 1$. Keluaran dari komponen $i$ ($0 \leq i \leq M$) terhubung dengan komponen `C[i]`.

- `X`, `Y`: array dengan panjang yang sama. Panjang $S$ dari array ini adalah banyaknya *switch* yang digunakan. Untuk *switch* $-j$ ($1 \le j \le S$), keluaran 'X' terhubung dengan komponen `X[j - 1]` dan keluaran 'Y' terhubung dengan komponen `Y[j - 1]`.
- Setiap elemen dari `C`, `X`, dan `Y` harus merupakan sebuah bilangan bulat di antara $-S$ dan $M$, inklusif.
- $S$ tidak lebih dari $400\,000$.
- Prosedur ini harus dipanggil tepat sekali.
- Sirkuit yang direpresentasikan dengan `C`, `X`, dan `Y` harus memenuhi kondisi pada deskripsi soal.

Apabila beberapa kondisi di atas tidak terpenuhi, program Anda akan dinilai sebagai **`Wrong Answer`**. Selain itu, program Anda akan dinilai sebagai **`Accepted`** dan nilai Anda akan dihitung berdasarkan $S$ (lihat subsoal).

# Contoh

Diberikan $M = 4$, $N = 4$, dan $A = [1, 2, 1, 3]$. Grader memanggil `create_circuit(4, [1, 2, 1, 3])`.




Gambar di atas menunjukkan sebuah sirkuit yang dideskripsikan dengan pemanggilan `answer([1, -1, -2, 0, 2], [2, -2], [3, 1])`. Angka pada gambar merupakan nomor seri dari setiap komponen.

Dua *switch* digunakan, sehingga $S = 2$.

Pada awalnya, *status* dari *switch* $-1$ dan $-2$ adalah 'X'.

Bola tersebut menelusuri sirkuit sebagai berikut:

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

- Ketika bola pertama kali memasuki *switch* $-1$, *status switch* tersebut adalah 'X'. Jadi, bola tersebut masuk ke *trigger* $2$. Kemudian *status* dari *switch* $-1$ berubah menjadi 'Y'.
- Ketika bola tersebut memasuki *switch* $-1$ untuk kedua kalinya, *status switch* tersebut adalah 'Y'. Jadi, bola tersebut masuk ke *trigger* $3$. Kemudian *status* dari *switch* $-1$ berubah menjadi 'X'.

Ketika bola tersebut pertama kali kembali ke *origin*, bola tersebut telah memasuki *trigger* $1, 2, 1, 3$ dan *status switch* $-1$ dan $-2$ adalah 'X'. Nilai dari $P$ adalah $4$. Oleh karena itu, sirkuit ini memenuhi kondisi di atas.

Berkas `sample-01-in.txt` di dalam zip lampiran merupakan masukan dari contoh ini. Contoh masukan lain juga tersedia di dalam paket tersebut.

# Batasan

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

# Subsoal

Nilai dan batasan untuk setiap subsoal adalah sebagai berikut:

1. (2 poin) Untuk setiap $i$ $(1 \le i \le M)$, bilangan bulat $i$ muncul paling banyak sekali di urutan $A_0, A_1, \ldots, A_{N-1}$.
2. (4 poin) Untuk setiap $i$ $(1 \le i \le M)$, bilangan bulat $i$ muncul paling banyak dua kali di urutan $A_0, A_1, \ldots, A_{N-1}$.
3. (10 poin) Untuk setiap $i$ $(1 \le i \le M)$, bilangan bulat $i$ muncul paling banyak 4 kali di urutan $A_0, A_1, \ldots, A_{N-1}$.
4. (10 poin) $N = 16$
5. (18 poin) $M = 1$
6. (56 poin) Tidak ada batasan khusus

Untuk setiap kasus uji, apabila program Anda dinilai sebagai **`Accepted`**, nilai Anda akan dihitung berdasarkan pada nilai $S$:

- Jika $S \le N + \log_2 N$, Anda mendapatkan nilai penuh untuk kasus uji tersebut.
- Untuk setiap kasus uji pada Subsoal 5 dan 6, Jika $N + \log_2 N < S \le 2N$, Anda mendapatkan sebagian nilai. Nilai untuk kasus uji tersebut adalah $0.5 + 0.4 \times \left(\dfrac{2N - S}{N - \log_2 N}\right)^2$, dikali dengan nilai yang diberikan pada subsoal tersebut.

- Selain itu, nilai Anda adalah 0.

Perhatikan bahwa nilai Anda untuk setiap subsoal adalah nilai minimum dari semua kasus uji yang ada di dalam subsoal tersebut.

# Contoh grader

Grader contoh membaca input dengan format sebagai berikut:

- baris 1: $M$ $N$
- baris 2: $A_0$ $A_1$ $\ldots$ $A_{N-1}$

Grader contoh akan mencetak tiga keluaran.

Pertama, grader contoh akan mencetak jawaban Anda pada sebuah berkas dengan nama `out.txt` dengan format sebagai berikut:

- baris 1: $S$
- baris $2 + i$ ($0 \le i \le M$): `C[i]`
- baris $2 + M + j$ ($1 \le j \le S$): `X[j - 1] Y[j - 1]`

Kedua, grader contoh akan mensimulasikan perjalanan bola. Grader contoh akan mencetak nomor seri dari komponen yang dilewati oleh bola tersebut pada sebuah berkas dengan nama `log.txt`.

Ketiga, grader contoh akan mencetak hasil evaluasi dari jawaban Anda pada standard output.

- Jika program Anda dinilai sebagai **`Accepted`**, grader contoh akan mencetak $S$ dan $P$ dengan format sebagai berikut `Accepted: S P`.

- Jika program Anda dinilai sebagai **`Wrong Answer`**, grader contoh akan mencetak `Wrong Answer: MSG`. dengan `MSG` adalah salah satu dari:

    - `answered not exactly once`: Prosedur `answer` tidak dipanggil tepat sekali.
    - `wrong array length`: Panjang dari `C` bukan $M + 1$, atau panjang dari `X` dan `Y` berbeda.
    - `over 400000 switches`: $S$ lebih besar dari $400\,000$.
    - `wrong serial number`: Terdapat elemen dari `C`, `X`, atau `Y` yang lebih kecil dari $-S$ atau lebih besar dari $M$.
    - `over 20000000 inversions`: Bola tidak kembali ke *origin* setelah $20\,000\,000$ perubahan *status* pada *switch*.
    - `state 'Y'`: Terdapat sebuah *switch* dengan *status* 'Y' setelah bola tersebut kembali ke *origin* untuk pertama kali.
    - `wrong motion`: *Trigger* yang mengakibatkan gerakan berbeda dengan urutan $A$.

Perhatikan bahwa grader contoh mungkin tidak akan membuat `out.txt` dan/atau `log.txt` ketika program Anda dinilai sebagai `Wrong Answer`.