მექანიკური თოჯინა

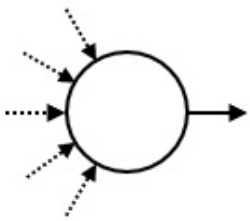
მექანიკური ეწოდება ისეთ თოჯინას, რომელიც იმეორებს სპეციფიკურ მოძრაობათა რაიმე მიმდევრობას. იაპონიაში უძველესი დროიდან იქმნებოდა სხვადასხვა მექანიკური თოჯინები.

მექანიკური თოჯინის მოძრაობები იმართება სპეციალური **სქემის** საშუალებით, რომელიც **მოწყობილობებისგან** შედგება. მოწყობილობები ერთმანეთთან დაკავშირებულია მილების საშუალებით. ყოველ მოწყობილობას აქვს ერთი ან ორი **გამოსასვლელი** და შესაძლოა ჰქონდეს ნებისმიერი რაოდენობის (შესაძლოა არცერთი) **შესასვლელები**. ყოველი მილი აერთებს მოწყობილობის გამოსასვლელს იგივე ან სხვა მოწყობილობის შესასვლელთან. ზუსტად ერთი მილი უერთდება თითოეულ შესასვლელს და ზუსტად ერთი მილი უერთდება თითოეულ გამოსასვლელს.

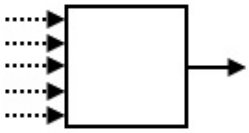
იმის აღსაწერად, თუ როგორ აკეთებს თოჯინა მოძრაობებს, განვიხილოთ **ბურთულა**, რომელიც მოთავსებულია ერთ-ერთ მოწყობილობაში. ბურთულა დაგორავს სქემაში. მოძრაობის ყოველ ბიჯზე ის ტოვებს მოწყობილობას მისი ერთ-ერთი გამოსასვლელიდან, მიგორავს ამ გამოსასვლელთან დაკავშირებულ მილში და შედის მოწყობილობაში ამ მილის მეორე ბოლოდან.

სქემაში სულ სამი სახის მოწყობილობა არსებობს: **საწყისი**, **ტრიგერი** და **გადამრთველი**. ამათგან საწყისი მოწყობილობა მხოლოდ ერთია, ტრიგერი M რაოდენობისაა, ხოლო გადამრთველთა რაოდენობა S-ის ტოლია (S შეიძლება ნულიც იყოს) და მისი მნიშვნელობა თქვენ უნდა განსაზღვროთ. ყველა მოწყობილობას თავისი უნიკალური სერიული ნომერი აქვს.

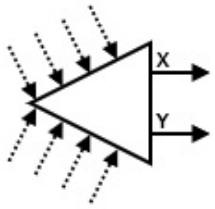
თავიდან ბურთულა საწყის მოწყობილობაშია მოთავსებული. მას მხოლოდ ერთი გამოსასვლელი აქვს და მისი სერიული ნომერი 0-ის ტოლია.



როდესაც ბურთულა შედის ტრიგერში, ის აიძულებს თოჯინას გააკეთოს ერთი სპეციფიკური მოძრაობა. ყოველ ტრიგერს აქვს მხოლოდ ერთი გამოსასვლელი და მისი სერიული ნომერი არის მთელი რიცხვი 1-დან M-მდე შუალედიდან.



ყოველ გადამრთველს აქვს ორი გამოსასვლელი, რომლებიც აღინიშნება 'X'-ით და 'Y'-ით. გადამრთველის **მდგომარეობა** არის 'X' ან 'Y'. ბურთულის გადამრთველში შესვლის შემდეგ იგი ტოვებს მას გამოსასვლელიდან, რომელიც გადამრთველის მიმდინარე მდგომარეობით არის მოცემული, რის შემდეგაც გადამრთველი მყისიერად იცვლის თავის მდგომარეობას საწინააღმდეგოთი. თავიდან ყველა გადამრთველი 'X' მდგომარეობაშია. გადამრთველთა სერიული ნომრები წარმოადგენს მთელ რიცხვებს -1-დან $-S$-მდე შუალედიდან.



თქვენ გეძლევათ ტრიგერთა რაოდენობა M. თქვენ მოცემული გაქვთ აგრეთვე N სიგრძის A მიმდევრობა, რომლის თითოეული ელემენტიც ტრიგერის სერიულ ნომერს წარმოადგენს. თითოეული ტრიგერი A მიმდევრობაში შეიძლება რამდენჯერმე (შეიძლება არცერთხელ) გვხვდებოდეს.

თქვენი ამოცანაა შექმნათ ისეთი სქემა, რომელიც დააკმაყოფილებს შემდეგ პირობებს:

- გარკვეული რაოდენობის ბიჯების შემდეგ ბურთულა ბრუნდება საწყის მოწყობილობაში.
- როდესაც ბურთულა პირველად ბრუნდება საწყის მოწყობილობაში, ყველა გადამრთველი უნდა იყოს 'X' მდგომარეობაში.
- ბურთულა პირველად ბრუნდება საწყის მოწყობილობაში ტრიგერების ზუსტად N რაოდენობის გავლის შემდეგ. მათი გავლის რიგითობის მიხედვით დალაგებული გავლილი ტრიგერების სერიული ნომრებია $A_0, A_1, \ldots, A_{N-1}$.
- ვთქვათ P არის ყველა იმ გადამრთველის მდგომარეობათა შეცვლის ჯამური რაოდენობა, რომელიც გამოიწვია ბურთულამ მის საწყის მოწყობილობაში პირველად დაბრუნებამდე. P-ს მნიშვნელობა არ აღემატება 20 000 000-ს.

ამავე დროს, თქვენი გსურთ რაც შეიძლება ნაკლები რაოდენობის გადამრთველის გამოყენება.

იმპლემენტაციის დეტალები

თქვენ უნდა მოახდინოთ შემდეგი პროცედურის იმპლემენტაცია.

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

- M: ტრიგერების რაოდენობა.
- A: მასივი სიგრძით N, რომლითაც მოცემულია მათი გავლის რიგითობის მიხედვით დალაგებული გავლილი ტრიგერების სერიული ნომრები.
- ეს პროცედურა გამოიძახება ზუსტად ერთხელ.
- შევნიშნოთ, რომ N წარმოადგენს A მასივის სიგრძეს და შეიძლება მიღებული იქნას ისე, როგორც იმპლემენტაციის აღწერაშია მითითებული.

თქვენმა პროგრამამ პასუხის მისაღებად უნდა გამოიძახოს შემდეგი პროცედურა:

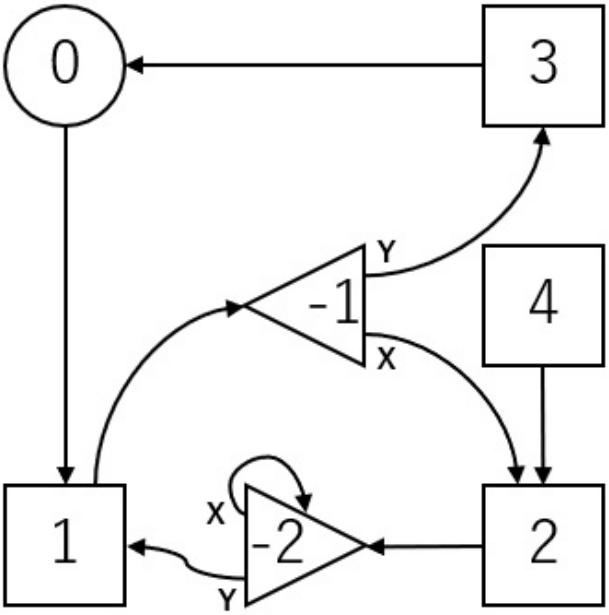
answer(int[] C, int[] X, int[] Y) ```

- C: მასივი სიგრძით $M + 1$. i-ური ($0 \leq i \leq M$) მოწყობილობის გამოსასვლელი შეერთებულია C[i] მოწყობილობასთან.
- X, Y: ტოლი სიგრძეების მქონე მასივები. ამ მასივთა S სიგრძე წარმოადგენს გადამრთველთა რაოდენობას. $-j$-ური ($1 \leq j \leq S$) გადამრთველის 'X' გამოსასვლელი დაკავშირებულია X[j - 1] მოწყობილობასთან, ხოლო 'Y' გამოსასვლელი დაკავშირებულია Y[j - 1] მოწყობილობასთან.
- C-ს, X-ის და Y-ის ყოველი ელემენტი უნდა იყოს მთელი რიცხვი $-S$-დან M-მდე (ჩათვლით).
- S არ უნდა აღემატებოდეს $400\,000$-ს.
- ეს პროცედურა გამოძახებული უნდა იქნას ზუსტად ერთხელ.
- C-თი, X-ით და Y-ით წარმოდგენილი სქემა უნდა აკმაყოფილებდეს ამოცანის პირობაში მოცემულ მოთხოვნებს.

თუ ზემოთ აღწერილი პირობებიდან რომელიმე არ სრულდება, თქვენი პროგრამის შეფასება იქნება **Wrong Answer**. წინააღმდეგ შემთხვევაში, თქვენი პროგრამის შეფასება იქნება **Accepted** და თქვენს მიერ მიღებული ქულები გამოითვლება S-ის მიხედვით (იხილეთ ქვეამოცანები).

მაგალითი

ვთქვათ $M = 4$, $N = 4$, and $A = [1, 2, 1, 3]$. გრადერი იძახებს create_circuit(4, [1, 2, 1, 3]).



ზემოთ მოცემული ნახაზი გვიჩვენებს სქემას, რომელიც აღწერილია `answer([1, -1, -2, 0, 2], [2, -2], [3, 1])`-ის გამოძახებით. სქემაზე ნაჩვენები რიცხვები მოწყობილობათა სერიულ ნომრებს წარმოადგენს.

გამოყენებულია ორი გადამრთველი, შესაბამისად $S = 2$.

თავიდან, ორივე -1 და -2 გადამრთველების მდგომარეობა არის 'X'.

ბურთულა მოძრაობს შემდეგნაირად:

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

- როდესაც ბურთულა პირველად შევა გადამრთველში -1, მისი მდგომარეობა იქნება 'X'. შესაბამისად, ბურთულა გადავა ტრიგერში 2, რის შემდეგაც ამ ტრიგერის მდგომარეობა შეიცვლება და გახდება 'Y'.
- როდესაც ბურთულა მეორედ შევა გადამრთველში -1, მისი მდგომარეობა უკვე იქნება 'Y'. შესაბამისად, ბურთულა გადავა ტრიგერში 3, რის შემდეგაც ამ გადამრთველის მდგომარეობა შეიცვლება და კვლავ გახდება 'X'.

ბურთულა პირველად დაბრუნდება საწყის მოწყობილობაში და გავლილი ექნება ტრიგერები $1, 2, 1, 3$. -1 და -2 გადამრთველებიდან ორივეს მდგომარეობა იქნება 'X'. P-ს მნიშვნელობა 4-ის ტოლია. შესაბამისად, სქემა აკმაყოფილებს პირობაში მოცემულ მოთხოვნებს.

დაარქივებულ მიბმულ პაკეტში არსებული `sample-01-in.txt` ფაილი წარმოადგენს მოცემული მაგალითის შესაბამის ფაილს. ამ პაკეტში ასევე ხელმისაწვდომია სხვა სანიმუშო შესატანი მონაცემები.

შეზღუდვები

- $1 \leq M \leq 100\,000$

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

ქვეამოცანები

ქულები და შეზღუდვები ყველა ტესტისათვის შემდეგია:

1. (2 ქულა) თითოეული i-სათვის ($1 \leq i \leq M$), მთელი i გვხვდება არაუმეტეს ერთხელ $A_0, A_1, \ldots, A_{N-1}$ მიმდევრობაში.
2. (4 ქულა) თითოეული i-სათვის ($1 \leq i \leq M$), მთელი i გვხვდება არაუმეტეს ორჯერ $A_0, A_1, \ldots, A_{N-1}$ მიმდევრობაში.
3. (10 ქულა) თითოეული i-სათვის ($1 \leq i \leq M$), მთელი i გვხვდება არაუმეტეს ოთხჯერ $A_0, A_1, \ldots, A_{N-1}$ მიმდევრობაში.
4. (10 ქულა) $N = 16$.
5. (18 ქულა) $M = 1$.
6. (56 ქულა) დამატებითი შეზღუდვების გარეშე.

თითოეულ ტესტში, რომელშიც თქვენი პროგრამა მიიღებს შეფასებას **Accepted**, თქვენს მიერ მიღებული ქულათა რაოდენობა გამოითვლება S-ის მნიშვნელობის მიხედვით შემდეგნაირად:

- თუ $S \leq N + \log_2 N$, თქვენ მიიღებთ სრულ ქულას ამ ტესტში.
- 5 და 6 ქვეამოცანების ყოველი ტესტისათვის, თუ $N + \log_2 N < S \leq 2N$, თქვენ მიიღებთ ნაწილობრივ ქულებს. ტესტში მიღებული ქულა ტოლია $0.5 + 0.4 \times \left(\frac{2N - S}{N - \log_2 N}\right)^2$ გამრავლებული ქვეამოცანაზე მინიჭებულ ქულაზე.
- სხვა შემთხვევაში თქვენ მიიღებთ 0 ქულას .

შევნიშნოთ, რომ თითოეულ ქვეამოცანაში თქვენს მიერ მიღებული ქულათა რაოდენობა წარმოადგენს მინიმალურს ამ ქვეამოცანაში თითოეული ტესტისათვის მიღებულ ქულებს შორის.

სანიმუშო გრადერი

სანიმუშო გრადერი შესატან მონაცემებს კითხულობს სტანდარტული შეტანიდან შემდეგ ფორმატში:

- სტრიქონი 1: M N
- სტრიქონი 2: A_0 A_1 … A_{N-1}

სანიმუშო გრადერი აწარმოებს სამ გამოსატან მონაცემს.

პირველ ყოვლისა, სანიმუშო გრადერს გამოაქვს თქვენი პასუხი ფაილში `out.txt` შემდეგი ფორმატით:

- სტრიქონი 1: S

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

შემდეგ სანიმუშო გრადერი სიმულაციას უკეთებს ბურთულის მოძრაობას. მას გამოაქვს იმ მოწყობილობების სერიული ნომრები, რომლებსაც გაივლის ბურთულა `log.txt` ფაილში მოცემული მიმდევრობით.

და ბოლოს, სანიმუშო გრადერი დაბეჭდავს თქვენი პასუხის შეფასებას სტანდარტულ გამოტანაში.

- თუ თქვენი პროგრამის შეფასებაა **Accepted**, მაშინ სანიმუშო გრადერი ბეჭდავს S-ს და P-ს შემდეგი ფორმატით: `Accepted: S P`.

- თუ თქვენი პროგრამის შეფასებაა **Wrong Answer**, იგი ბეჭდავს `Wrong Answer: MSG`. სადაც `MSG` ნიშნავს შემდეგს:

    - `answered not exactly once`: პროცედურა `answer` არ იქნა გამოძახებული ზუსტად ერთხელ.
    - `wrong array length`: `C`-ს სიგრძე არ არის $M + 1$, ან `X`-ის და `Y`-ის სიგრძეები განსხვავებულია.
    - `over 400000 switches`: S აღემატება $400\,000$-ს.
    - `wrong serial number`: არსებობს `C`-ს, `X`-ის ან `Y`-ის ისეთი ელემენტი, რომელიც ნაკლებია $-S$-ზე ან მეტია M-ზე.
    - `over 20000000 inversions`: ბურთულა არ ბრუნდება საწყის მოწყობილობაში გადამრთველთა მდგომარეობების $20\,000\,000$-ჯერ შეცვლის შემდეგ.
    - `state 'Y'`: ბურთულის საწყის მოწყობილობაში პირველად დაბრუნების შემდეგ არსებობს გადამრთველი, რომლის მდგომარეობაც არის `'Y'`.
    - `wrong motion`: ტრიგერები, რომლებიც იწვევენ თოჯინის მოძრაობას, განსხვავებულია A მიმდევრობისაგან.

შევნიშნოთ, რომ სანიმუშო გრადერმა შეიძლება არ შექმნას `out.txt` და/ან `log.txt`, როცა თქვენი პროგრამის შეფასება იქნება `Wrong Answer`.