Aqua Racer (A)

Limit pamięci: 1024 MB Limit czasu: 1.00 s

W grze *Aqua Racer* wcielasz się w sternika supernowoczesnej motorówki, pędzącej po wodnym torze przeszkód. Jednak nie jest to zwykły wyścig — tutaj liczy się nie tylko szybkość, ale też matematyczny refleks!

Podczas wyścigu trasa kilkukrotnie rozwidla się w trzy różne strony. Nad każdą z dalszych części trasy pojawia się inna liczba. Tuż przed rozgałęzieniem zostaje także ujawnione pewne wyrażenie matematyczne. Dokładnie jedna z liczb jest jego wynikiem. Gracz musi szybko obliczyć jego wynik i skierować swoją motorówkę w odpowiednią stronę, aby nie wypaść z trasy.

Twoim zadaniem jest napisanie programu-sterownika do gry *Aqua Racer*, który:

- otrzymuje jako wejście trzy dodatnie liczby całkowite,
- wypisuje proste wyrażenie arytmetyczne złożone z dwóch dodatnich liczb całkowitych oraz jednego znaku spośród: `+`, `-` oraz `*`,
- wynik tego wyrażenia musi być równy dokładnie jednej z trzech podanych liczb,
- jeśli nie da się znaleźć takiego wyrażenia — wypisz zamiast tego jedno słowo: `NIE`.

Wejście

W pierwszym i jedynym wierszu wejścia podane są trzy dodatnie liczby całkowite A, B oraz C, oznaczające trzy dane liczby, dla których należy znaleźć odpowiednie wyrażenie.

Wyjście

W pierwszym i jedynym wierszu wyjścia należy wypisać wyrażenie arytmetyczne w formacie `X znak Y`, gdzie `znak` jest jednym z symboli `+`, `-` albo `*`, a `X` oraz `Y` są liczbami całkowitymi z przedziału od 1 do $2 \cdot 10^9$ włącznie. Znak powinien być oddzielony od każdej z liczb pojedynczym odstępem. Jeśli nie ma takiego wyrażenia, zamiast tego wypisz jedno słowo `NIE`.

Ograniczenia

$1 \leq A, B, C \leq 10^9$.

Przykłady

Wejście	Wyjście
`3 7 15`	`3 + 4`
`6 3 14`	`5 - 2`
`4 12 1`	`3 * 4`

Wejście	Wyjście
5 5 5	NIE

Mistrzostwa Polski Szkół Średnich w Programowaniu Zespołowym

Kraken (B)

Limit pamięci: 1024 MB Limit czasu: 5.00 s

Strzeżcie się, morscy żeglarze! Jasio Binarnobrody, postrach wszelkich mórz i oceanów, wyrusza właśnie na kolejne pirackie podboje. Już widzi następną wyspę skarbów, gdy nagle z morskiej toni wynurza się Kraken – ogromny stwór żywiący się ciałami zagubionych marynarzy. Jedyna droga naprzód prowadzi obok potwora, trzeba go więc pokonać.

Jak to każde pirackie dziecko wie, Kraken ma mnóstwo macek, a na każdej pewną liczbę przyssawek. By pokonać Krakena, *xor* liczb przyssawek na mackach, które znajdują się ponad powierzchnią wody, musi być równy ich *and*-owi. Binarnobrody to waleczny pirat, więc gdy zobaczy, że nie może aktualnie pokonać Krakena, zdecyduje się strzelić ze swojej armaty, by pozbyć się wybranej przez siebie macki. Musi on być jednak ostrożny, gdyż może wykonać tylko jeden strzał!

Binarnobrody zauważył, że na początku żadna macka Krakena nie wystawała z wody. W i-tej chwili potwór wykona jedną z dwóch akcji:

- wynurzy mackę o liczbie przyssawek x_i (operacja + x_i),
- lub zanurzy jedną z wyłonionych wcześniej macek o liczbie przyssawek x_i (operacja - x_i).

Twoim zadaniem jest – po każdej operacji – odpowiedzieć, czy Binarnobrody mógłby pokonać Krakena, po ewentualnym (co najwyżej jednym) strzale z armaty w wybraną przez siebie mackę.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba całkowita, będąca liczbą zmian, oznaczana przez Q.
W i-tym z następnych Q wierszy znajduje się albo znak + albo - oraz liczba x_i oznaczająca odpowiednio wynurzenie się macki o liczbie przyssawek równej x_i albo jej zanurzenie. Gwarantowane jest, że w chwili, w której macka o danej liczbie przyssawek miałaby zostać zanurzona, jest wynurzona przynajmniej jedna macka o takiej liczbie przyssawek.

Wyjście

Wyjście powinno składać się z Q wierszy.
W i-tym z nich należy umieścić słowo `TAK`, jeżeli w i-tej chwili, po zmianie, można wyrównać *xor* i *and* liczb przyssawek, pozbywając się co najwyżej jednej macki. W przeciwnym wypadku, w i-tym wierszu należy umieścić słowo `NIE`.

Ograniczenia

$1 \leq Q \leq 1\,000\,000$, $0 \leq x_i \leq 1\,000\,000$.
Zarówno *xor* jak i *and* zbioru pustego definiujemy jako zero.

Przykłady

Wejście	Wyjście	Wyjaśnienie
5 + 10 + 8 + 3 + 10 - 8	TAK TAK NIE TAK NIE	Po czwartej operacji Binarnobrody może pokonać Krakena, jeżeli ustrzeli mackę z 3 przyssawkami. Wtedy *xor* i *and* przyssawek na pozostałych mackach są sobie równe ($10\ xor\ 8\ xor\ 10 = 8 = 10\ and\ 8\ and\ 10$).

Kąpiel (C)

Limit pamięci: 1024 MB Limit czasu: 1.00 s

Po ciężkim dniu pracy Jasio postanowił się zrelaksować, biorąc kąpiel. Aby osiągnąć idealny komfort, chce napełnić wannę **co najmniej** v litrami wody o temperaturze **dokładnie** t stopni Bajtocjusza.

Przy wannie znajduje się kran, z którego leci woda z prędkością 1 litra na sekundę. Ze względu na swoje ekologiczne podejście, Jasio nie może wylewać wody w trakcie napełniania – cała woda z kranu musi trafić do wanny.

Temperatura początkowa wody wynosi z stopni Bajtocjusza. Aby ją zmienić, Jasio może w dowolnym **rzeczywistym** momencie włączać i wyłączać bojler:

- Gdy bojler jest włączony, temperatura wody rośnie o 1 stopień na sekundę, aż osiągnie wartość maksymalną c, po czym pozostaje stała.
- Gdy bojler jest wyłączony, temperatura wody spada o 1 stopień na sekundę, aż osiągnie wartość minimalną z, po czym także pozostaje stała.

Mieszanie cieczy zachowuje prawa fizyki. Formalnie, łącząc l_1 litrów wody o temperaturze t_1 z l_2 litrami wody o temperaturze t_2, otrzymujemy $(l_1 + l_2)$ litrów wody o temperaturze $\frac{t_1 l_1 + t_2 l_2}{l_1 + l_2}$.

Temperatura wody zmienia się tylko w trakcie lania, dlatego Jasio nigdy nie zakręca kranu podczas napełniania wanny. Co więcej, woda w wannie stygnie na tyle wolno, że Jasio nawet nie jest w stanie zauważyć tego faktu.

Twoim zadaniem jest określić, ile najmniej czasu potrzeba, aby napełnić wannę co najmniej v litrami wody o temperaturze dokładnie t stopni.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba całkowita T, oznaczająca liczbę przypadków testowych. W kolejnych T wierszach znajdują się opisy tych przypadków.
Każdy przypadek testowy składa się z czterech liczb całkowitych c, z, t oraz v. Oznaczają one kolejno maksymalną, minimalną (a zarazem początkową) i oczekiwaną na końcu temperaturę wody oraz minimalną końcową ilość wody.

Wyjście

Dla każdego przypadku testowego wypisz na wyjściu jeden wiersz zawierający jedną liczbę rzeczywistą, oznaczającą minimalny czas lania wody (w sekundach) potrzebny do uzyskania co najmniej v litrów wody o temperaturze dokładnie t stopni Bajtocjusza.
Jeżeli w którymś z przypadków nie da się uzyskać podanej temperatury wody w **skończonym** czasie, dla danego przypadku wypisz wartość -1. Wszystkie Twoje odpowiedzi zostaną uznane za poprawne jeśli ich błąd względny bądź bezwzględny będzie nie większy niż 10^{-6}.

Ograniczenia

$1 \leq T, t \leq 100$, $1 \leq z < c \leq 100$, $1 \leq v \leq 10^6$.

Przykłady

Wejście	Wyjście
3 80 20 40 30 75 15 70 50 77 13 90 1	40.0000000000 360.0000000000 -1

Misja Namazu (D)

Limit pamięci: 1024 MB Limit czasu: 3.00 s

Jan jest naukowcem w instytucji oceanograficznej. Wraz z całym zespołem sejsmologów pełni bardzo odpowiedzialne zadanie – monitoruje podwodne wstrząsy ziemi, aby jak najdokładniej przewidywać powstawanie zagrażających ludziom fal.

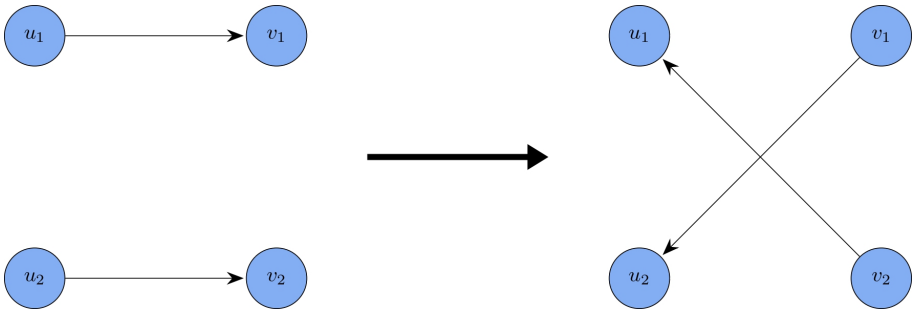
W ostatnim czasie do floty urządzeń badających dno oceanu dołączył nowy sejsmograf. Niestety, informacje przez niego wysyłane od początku wyglądały na zupełnie losowy ciąg bajtów – wszystko wskazuje na to, że sejsmograf ma poważną usterkę montażową. Do błędu przyznał się młody, niedoświadczony członek zespołu konstrukcyjnego. Jak się okazało, pomylił on schematy połączeń przewodów do transmisji danych.

Na szczęście Jan ma pod ręką zarówno schemat, według którego maszyna jest obecnie podłączona, jak i poprawny schemat, który mógłby sprawić, że urządzenie zacznie działać. Schematy przedstawiają N modułów oraz łączące je przewody światłowodowe, które potrafią przesyłać informacje tylko w jedną stronę. Na obu schematach z każdego modułu wychodzi dokładnie jeden przewód oraz do każdego modułu wchodzi dokładnie jeden przewód. Przewody mogą wchodzić do tego samego modułu, z którego wychodzą.

Jedyne, co pozostało to naprawa błędu, czyli odpowiednie przepięcie przewodów. Ponieważ sejsmograf jest już zakotwiczony na dnie, wydobycie go na powierzchnię nie wchodzi w grę. Jedyną opcją jest wysłanie na dno robota, który wykona to zadanie. Niestety, Jan ma do dyspozycji tylko jednego robota głębinowego – Namazu, który w dodatku jest dość przestarzały.

Ograniczenia Namazu są dość uciążliwe. Po pierwsze, bardzo trudno komunikować się z nim na odległość, zatem całą sekwencję ruchów, które ma wykonać, musi mieć zaprogramowaną jeszcze przed umieszczeniem w wodzie. Po drugie, potrafi wykonywać operacje tylko jednego typu, którą można opisać czterema (niekoniecznie różnymi) liczbami u_1, v_1, u_2, v_2, będącymi numerami modułów. Przebiega ona następująco:

- Sprawdź, czy istnieją dwa różne kable prowadzące z u_1 do v_1 i z u_2 do v_2. Jeżeli nie, zwróć `ERROR` (wtedy Namazu się zawiesza).
- Odłącz przewody prowadzące z u_1 do v_1 oraz z u_2 do v_2.
- Poprowadź światłowód z v_1 do u_2 i z v_2 do u_1.



Na rysunku przedstawiono sposób w jaki Namazu przepnie przewody, jeżeli wykona instrukcję $u_1\ v_1\ u_2\ v_2$.

Jan nie ma pewności co do tego, czy Namazu jest w stanie naprawić sejsmograf. Nawet jeśli jest to możliwe, to zaprogramowanie ciągu instrukcji w pamięci robota znacząco wykracza poza kompetencje sejsmologa. Dlatego poprosił Cię o pomoc. Mając dane dwa schematy – aktualny oraz docelowy – określ, czy robot jest w stanie naprawić sejsmograf. Jeśli tak, wypisz również ciąg czteroliczbowych instrukcji, które sprawią, że przewody będą podłączone tak jak na schemacie docelowym, natomiast robot się nie zawiesi.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba naturalna N oznaczająca liczbę modułów w sejsmografie.
W drugim wierszu wejścia znajduje się ciąg N liczb a_i, opisujący aktualny sposób podłączenia kabli w urządzeniu. Liczba a_i oznacza, że jest poprowadzony światłowód z i-tego do a_i-tego modułu.
W trzecim wierszu wejścia znajduje się ciąg N liczb b_i, opisujący poprawny sposób podłączenia kabli w sejsmografie. Liczba b_i oznacza, że powinien zostać poprowadzony światłowód z i-tego do b_i-tego modułu.
W obu opisach zagwarantowane jest, że każdy moduł wystąpi w nim dokładnie raz.

Wyjście

W pierwszym wierszu wyjścia powinno znaleźć się jedno słowo – `TAK` jeśli istnieje ciąg instrukcji, które wykonane przez Namazu naprawią sejsmograf, albo `NIE`, w przeciwnym wypadku.
Jeżeli w pierwszym wierszu znalazło się słowo `TAK`, w następnych wierszach powinien znaleźć się opis instrukcji, które robot ma wykonać, aby naprawić urządzenie. W pierwszym wierszu opisu powinna znaleźć się jedna nieujemna liczba całkowita K oznaczająca długość ciągu instrukcji. Ciąg ten nie musi być minimalnej długości, ale nie może mieć więcej niż $1\,000\,000$ instrukcji.
W kolejnych K wierszach powinien znaleźć się opis kolejnych instrukcji wykonywanych przez robota, po jednej instrukcji w wierszu. Opis jednej instrukcji składa się z czterech liczb naturalnych u_1, v_1, u_2, v_2, oznaczających numery modułów, dla których robot powinien zmienić okablowanie. Namazu, wykonując taką instrukcję, odepnie kable prowadzące z modułu u_1 do modułu v_1 oraz z u_2 do v_2, a następnie poprowadzi przewody z v_1 do u_2 oraz z v_2 do u_1.
Jeżeli istnieje wiele poprawnych rozwiązań, możesz wypisać dowolne z nich.

Ograniczenia

$2 \leq N \leq 200\,000$, $1 \leq a_i, b_i \leq N$.

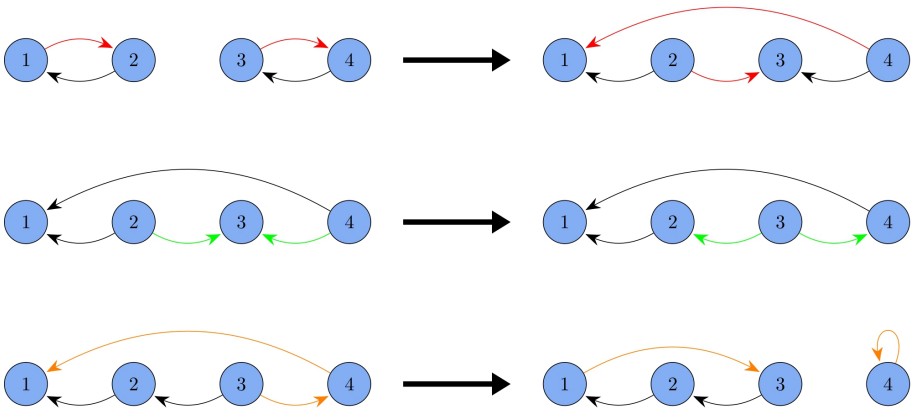
Przykłady do tego zadania znajdują się na osobnej kartce.

Przykłady (zadanie *Misja Namazu*)

Wejście
```
4
2 1 4 3
3 1 2 4
```

Wyjście
```
TAK
3
1 2 3 4
2 3 4 3
3 4 4 1
```

Wyjaśnienie
Sejsmograf z powyższego testu przykładowego można naprawić trzema instrukcjami. Poniżej przedstawiono jeden z takich ciągów instrukcji.



Wejście
```
2
1 2
1 2
```

Wyjście
```
TAK
0
```

Wejście
```
3
2 3 1
3 2 1
```

Wyjście
```
NIE
```

Wyjaśnienie
Można pokazać, że nie istnieje ciąg instrukcji dla Namazu, który naprawia sejsmograf z tego testu przykładowego.

Mistrzostwa Polski Szkół Średnich w Programowaniu Zespołowym

Zatopione zadania (E)

Limit pamięci: 1024 MB Limit czasu: 3.00 s

Podczas przygotowań do Mistrzostw Polski Szkół Średnich w Nurkowaniu Zespołowym, organizatorzy postanowili ukryć zadania w podwodnej jaskini. Jasio, bardzo zdeterminowany, by po raz kolejny przynieść chwałę swojej szkole nurkowania, postanowił wykraść mapę prowadzącą do tej jaskini.

Od znajomego dowiedział się, że mapa tworzona była w bardzo nietypowy sposób:

1. Mapa to ogromna kartka papieru, którą można traktować jako układ współrzędnych.
2. Organizatorzy wielokrotnie zginają ją wzdłuż prostych linii równoległych do osi współrzędnych w miejscu, w którym aktualnie jest **najwięcej** warstw.
3. Po serii zgięć wykonują jedną, niewielką dziurę – przebijając **wszystkie** warstwy złożonej mapy w miejscu gdzie jest ich najwięcej. Dziura ta nie pokrywa się z żadną linią zgięć.
4. Następnie mapę całkowicie rozkładają.

Na mapie, którą Jasio zdobył, widać dziury – wszystkie znajdują się w punktach kratowych. Zanim jednak wyruszy na poszukiwanie jaskini, musi upewnić się, że nie ukradł przypadkiem czegoś zupełnie nieprzydatnego.

Twoim zadaniem jest, na podstawie listy punktów na rozłożonej mapie, określić czy mogły one powstać w opisanym powyżej procesie.

Wejście

W pierwszym wierszu wejścia znajduje się liczba N oznaczająca liczbę punktów na mapie.
W kolejnych N wierszach znajdują się po dwie liczby, x_i i y_i, oznaczające współrzędne dziury na mapie. Żadna para punktów nie pokrywa się.

Wyjście

W pierwszym i jedynym wierszu wyjścia powinno znaleźć się jedno słowo `TAK`, jeżeli da się złożyć mapę, zrobić w niej jedną dziurę i rozłożyć tak, by podane punkty pokrywały się z dziurami. W przeciwnym wypadku na wyjściu powinno znaleźć się jedno słowo `NIE`.

Ograniczenia

$1 \leq N \leq 1\,000\,000$, $-10^{18} \leq x_i, y_i \leq 10^{18}$.

Przykłady

Wejście	Wyjście
8 -2 -2 -2 0 -2 4 -2 6 1 -2 1 0 1 4 1 6	TAK

Wejście	Wyjście
3 0 0 0 1 0 2	NIE

Zadanie od Jasia (F)

Limit pamięci: 1024 MB Limit czasu: 6.00 s

Jasio, po zakończeniu swoich startów w Olimpiadzie Informatycznej, zainteresował się eksploracją głębi oceanu. Ze stypendium, które otrzymał, zakupił nowy batyskaf i wybrał się na pierwszą głębinową wyprawę. Oczywiście, nawet podczas ekspedycji nie mógł zrezygnować z wymieniania się z przyjaciółmi ciekawymi zadaniami algorytmicznymi. Komunikacja z dna oceanu jest dosyć ograniczona, więc Jasio wysyła treści w jak najprostszym formacie, bez rozbudowanych historyjek. Oto jedna z nich:

Dany jest ciąg N liczb $a_1, \ldots, a_N$ oraz operacja *absolutnego zmniejszenia*, polegającą na zmianie wszystkich a_i o indeksach należących do wybranego przedziału $[l, r]$ na $|a_i - 1|$ (wartość bezwzględną z $a_i - 1$). Otrzymujemy Q zapytań. Każde zapytanie jest jednego z dwóch typów:

1. Dane są pozycja x oraz wartość y. Ustaw a_x na y.
2. Dany jest przedział od l do r włącznie. Podaj minimalną liczbę operacji *absolutnego zmniejszenia* potrzebną do zamienienia na zero wszystkich a_i dla i z przedziału $[l, r]$.

Twoim zadaniem jest napisać program obsługujący te zapytania.

Wejście

W pierwszym wierszu wejścia znajdują się dwie liczby całkowite N i Q, oznaczające kolejno długość ciągu i liczbę zapytań.
W drugim wierszu znajduje się ciąg $a_1, \ldots, a_N$, składający się z N liczb całkowitych.
W kolejnych Q wierszach znajduje się opis zapytań. W i-tym z nich znajdują się trzy liczby całkowite:

- $1\ x_i\ y_i$ – i-te zapytanie jest typu pierwszego z parametrami x_i, y_i.
- $2\ l_i\ r_i$ – i-te zapytanie jest typu drugiego z parametrami l_i, r_i.

Wyjście

Wyjście powinno składać się z tylu wierszy, ile pojawiło się zapytań typu drugiego. W i-tym z nich należy umieścić odpowiedź na i-te zapytanie typu drugiego.

Ograniczenia

$1 \leq N \leq 300\,000$, $1 \leq Q \leq 300\,000$, $0 \leq a_i, y_i \leq 10^9$, $1 \leq x_i \leq N$, $1 \leq l_i \leq r_i \leq N$.

Przykłady

Wejście	Wyjście
5 11 2 2 1 4 2 2 1 5 1 3 5 2 1 5 2 1 3 1 5 0 2 5 5 2 2 4 1 1 5 2 1 4 1 3 3 2 1 5	4 5 5 0 5 6 6

Podmorska świątynia (G)

Limit pamięci: 1024 MB Limit czasu: 3.00 s

Po przygodach z Krakenem, Jaś Binarnobrody udał się na wyspę piratów zwaną *Sortuga*, gdzie razem ze swoją załogą wydał wszystkie złupione kosztowności. Za ostatniego złotego talara kupił od nieznajomego mapę owianej legendami podmorskiej świątyni oraz kilka rurek do nurkowania.

Zgodnie z pozyskaną mapą, w świątyni znajduje się N komnat, ponumerowanych kolejnymi liczbami naturalnymi od 1 do N, połączonych siecią M korytarzy. Wejście do świątyni prowadzi prosto do komnaty o numerze 1. Celem wyprawy jest dobrnięcie do komnaty o numerze N, w której zgromadzone są skarby.

Legendy głoszą, że w niektórych komnatach znajdują się potwory. Do takich komnat nie będzie można wejść, ani przez nie przejść, dopóki nie pokona się stacjonującego w niej potwora. Nie wszystkie walki będą tak samo trudne, strażnicy świątyni różnią się siłą swojego ataku, którą można opisać liczbą naturalną z zakresu od 1 od N. Mapa wskazuje w których komnatach znajdują się potwory i ile wynoszą siły ich ataków.

Jaś Binarnobrody wie, że jego załoga musi nabrać doświadczenia w walce z potworami, nim porwie się na jakąś potężną bestię. Po każdej wygranej walce załoga Jasia zyskuje jeden punkt doświadczenia, lecz każda walka kosztuje życie jednego członka załogi. By pokonać potwora o sile ataku równej k, załoga musi posiadać przynajmniej k punktów doświadczenia. Jednakże, jeśli liczebność załogi spadnie do zera, to wyprawa zakończy się niepowodzeniem. Drużyna zaczyna mając 1 punkt doświadczenia.

Wejście do komnaty kończy się walką tylko wtedy, gdy jest w niej jakiś potwór. Przechodzenie po komnatach, w których potwora nie ma lub już został zabity, nie uśmierca żadnego członka załogi, ale również nie zwiększa doświadczenia.

Jaś zastanawia się teraz, ilu piratów co najmniej powinna liczyć cała jego załoga, aby udało im się dotrzeć do skarbów w komnacie o numerze N. Napisz program, który wczyta opis świątyni i wypisze odpowiedź na jego pytanie.

Wejście

W pierwszym wierszu wejścia znajdują się dwie liczby całkowite N, M oddzielone pojedynczym odstępem.
W drugim wierszu wejścia znajduje się N liczb całkowitych k_i pooddzielanych pojedynczymi odstępami, gdzie i-ta z nich oznacza siłę ataku potwora znajdującego się w i-tej komnacie. Jeśli liczba ta jest równa 0, to w i-tej komnacie nie ma potwora.
W kolejnych M wierszach wejścia znajdują się po dwie liczby całkowite a_i, b_i, oddzielone pojedynczym odstępem, oznaczające że istnieje korytarz pomiędzy komnatami o numerach a_i oraz b_i.
Możesz założyć, że sieć komnat tworzy graf spójny, pomiędzy żadną parą komnat nie istnieje więcej niż jeden korytarz, a w komnacie o numerze 1 nie znajduje się potwór.

Wyjście

W pierwszym i jedynym wierszu wyjścia należy wypisać jedną liczbę całkowitą będącą najmniejszą liczbą załogantów potrzebną do przeprowadzenia skutecznej wyprawy.
Jeżeli skuteczna wyprawa jest niemożliwa do przeprowadzenia, na wyjściu wypisz `-1`.

Ograniczenia

$2 \leq N \leq 300\,000$, $N - 1 \leq M \leq 300\,000$, $0 \leq k_i \leq N$, $1 \leq a_i, b_i \leq N$, $a_i \neq b_i$.

Przykłady

Wejście

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

Wyjście

```
5
```

Wejście

```
6 8
0 2 1 3 2 3
5 2
1 5
6 1
1 3
4 2
5 6
6 3
2 3
```

Wyjście

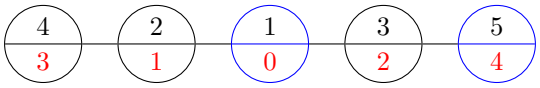
```
4
```

Wejście

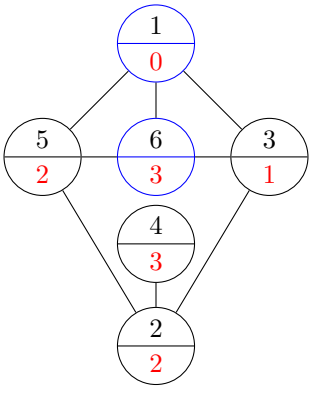
```
4 4
0 1 1 4
1 2
1 3
2 4
3 4
```

Wyjście

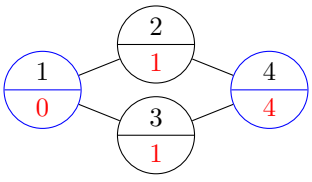
```
-1
```



Test przykładowy nr 1



Test przykładowy nr 2



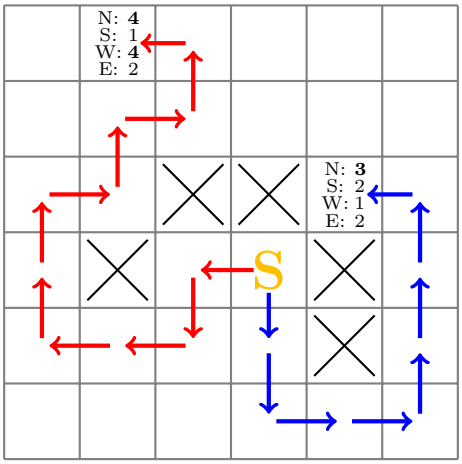
Test przykładowy nr 3

Mistrzostwa Polski Szkół Średnich w Programowaniu Zespołowym

Podwodne tankowanie (H)

Limit pamięci: 1024 MB Limit czasu: 10.00 s

Jasio zwiedza kwadratowy akwen, składający się z $N \times N$ sektorów, swoją nową łodzią podwodną. Posiada ona 4 silniki odpowiedzialne za 4 kierunki pływania: na północ, na południe, na wschód oraz na zachód.



Jasio potrzebuje zatankować swoje silniki (każdy silnik ma osobny zbiornik paliwa), ale nie wie jeszcze gdzie popłynie. Użycie dowolnego z silników kosztuje 1 litr paliwa z jego zbiornika oraz powoduje przemieszczenie się do sąsiedniego sektora.

Przed każdą wyprawą Jasio wlewa do wszystkich zbiorników dokładnie po K litrów paliwa. Oznacza to, że każdego silnika może użyć do poruszenia się w odpowiadającym mu kierunku nie więcej niż K razy.

Jasiowi udało się zdobyć mapę sonarową zbiornika. Niektóre z sektorów są na niej oznaczone jako zablokowane – nie można do nich wpływać. Dodatkowo warto pamiętać, że nie wolno wypływać poza $N \times N$ sektorów przedstawionych na mapie, gdyż nigdy nie wiadomo jakie czyhają tam niebezpieczeństwa!

Jedyne co pozostało Jasiowi, to policzyć ile minimalnie litrów paliwa potrzebuje wlać do wszystkich silników, w zależności od celu wyprawy. Ze względu na zmęczenie, o tę ostatnią część przygotowań poprosił Ciebie.

Pomóż Jasiowi i dla każdego z sektorów podaj minimalną liczbę K, która umożliwi dopłyniecie do tego miejsca z sektora startowego.

Wejście

W pierwszym wierszu znajduje się liczba naturalna N opisująca wielkość boku akwenu.
W kolejnych N wierszach znajduje się opis akwenu.
W $(i+1)$-szym wierszu na j-tej pozycji znajduje się pojedynczy znak kropki (`.`), gdy w sektorze (i, j) jest wolna przestrzeń, znak # gdy sektor jest zablokowany, bądź litera `S`, co oznacza pozycję startową łodzi podwodnej Jasia (sektor startowy na pewno nie jest zablokowany). Znak `S` występuje na wejściu dokładnie jeden raz.

Wyjście

Na wyjściu należy wypisać N wierszy, a w każdym z nich N liczb, oznaczających odpowiedzi dla wszystkich sektorów.
Na pozycji j w i-tym wierszu powinno znaleźć się minimalne K, takie że jeśli Jasio zatankuje swoje silniki do K litrów paliwa, będzie istniała sekwencja włączająca silniki, która pozwoli na dopłynięcie do sektora (i, j) z sektora startowego.
Jeżeli takie K nie istnieje (sektor jest nieosiągalny lub zablokowany), należy zamiast tego wypisać `-1`.

Ograniczenia

$1 \leq N \leq 90$.

Przykłady

Wejście	Wyjście
5 ....# ....S.. .###.	2 2 2 2 -1 2 1 1 1 2 2 1 0 1 2 2 -1 -1 -1 2 2 2 2 2 2

Wejście	Wyjście
6 ....##.. .#.S#. ....#.	4 4 4 4 4 5 3 3 3 3 4 4 3 3 -1 -1 3 3 3 -1 1 0 -1 2 3 2 1 1 -1 2 3 2 2 2 2 2

Naszyjnik z pereł (I)

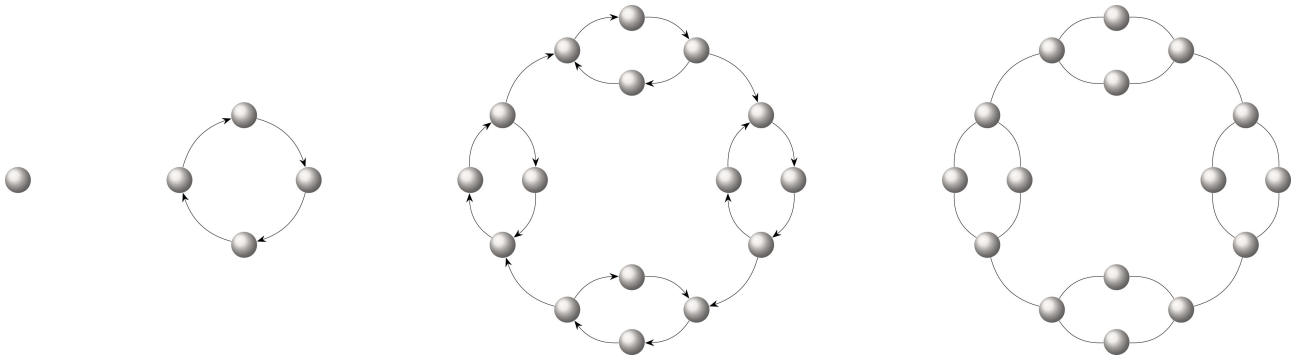
Limit pamięci: 1024 MB Limit czasu: 1.00 s

Syrenka Ariel wybiera się na wielki bal podwodnego królestwa. Aby wyglądać olśniewająco, postanowiła na tę okazję przygotować sobie wyjątkową ozdobę – naszyjnik z pereł.

Jako córka króla mórz, Arielka może stworzyć naszyjnik w magiczny sposób. Na początku ma ona przed sobą jedną perłę. Następnie, może ona rzucić N zaklęć $s_1, s_2, ..., s_N$. Rzucenie zaklęcia s_i powoduje, że każda perła naszyjnika zostaje zastąpiona przez cykliczny łańcuszek **parzystej** długości złożony z s_i pereł.

Dokładniej rzecz biorąc, proces tworzenia naszyjnika wygląda następująco:

- Stan naszyjnika przed rozpoczęciem czarów, jak i po rzuceniu każdego zaklęcia, można reprezentować poprzez pewien graf **skierowany**, którego wierzchołkami są perły.
- Przed rzuceniem pierwszego zaklęcia naszyjnik jest pojedynczą perłą, a więc grafem G_0 złożonym z jednego wierzchołka, bez żadnych krawędzi.
- G_i powstaje przez rzucenie zaklęcia s_i na graf G_{i-1}, a więc przez zastąpienie każdego wierzchołka u w grafie G_{i-1} cyklem złożonym z nowych wierzchołków $u_0, u_1, ..., u_{s_i-1}$. W G_i istnieją krawędzie z u_0 do u_1, z u_1 do u_2, ..., z u_{s_i-2} do u_{s_i-1} oraz z u_{s_i-1} do u_0. Wszystkie krawędzie wchodzące do u w G_{i-1} wchodzą w G_i do u_0, a wszystkie krawędzie wychodzące z u w G_{i-1} wychodzą w G_i z wierzchołka $u_{s_i/2}$.
- Po rzuceniu wszystkich zaklęć skierowanie krawędzi nie ma już znaczenia, więc każda krawędź skierowana z u do v jest zamieniana na krawędź nieskierowaną pomiędzy tymi samymi wierzchołkami.



Na rysunku przedstawiono przykładowy proces powstawania naszyjnika z użyciem zaklęć $4, 4$.

Morska Wiedźma Urszula jest zazdrosna o piękno księżniczki i planuje zniszczyć jej cenną ozdobę. Przetnie ona największą możliwą liczbę nici między perłami w taki sposób, aby naszyjnik nadal stanowił jedną spójną całość, tzn. aby każde dwie perły były połączone jakimś ciągiem nici. W przeciwnym wypadku Arielka zobaczyłaby rozsypane perły i zdążyłaby naprawić swe dzieło przed balem. Jedyna nadzieja w Tobie – jeżeli podasz Urszuli liczbę różnych sposobów, na które może to zrobić, być może ogrom możliwości przytłoczy ją i nie zdąży podjąć decyzji przed rozpoczęciem balu. Pomóż królewnie mórz w tarapatach!

Napisz program, który wczyta ciąg zaklęć rzuconych przez Ariel, oraz wypisze liczbę sposobów, na które Urszula może przeciąć najwięcej nici naszyjnika tak, aby pozostał on spójny, modulo $10^9 + 7$. Dwa sposoby przecięcia uważamy za różne, jeśli istnieje nić, która jest przecięta w jednym ze sposobów, a w drugim nie.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba całkowita N, oznaczająca liczbę zaklęć, których Ariel użyła do stworzenia naszyjnika.

W drugim wierszu znajduje się N **parzystych** liczb naturalnych s_i, pooddzielanych pojedynczymi odstępami. Oznaczają one kolejne zaklęcia rzucone przez Syrenkę.

Wyjście

W pierwszym i jedynym wierszu wejścia powinna znaleźć się jedna liczba całkowita oznaczająca liczbę możliwości zniszczenia naszyjnika, modulo $10^9 + 7$.

Ograniczenia

$1 \leq N \leq 100\,000$, $2 \leq s_i \leq 10^9$.

Przykłady

Wejście	**Wyjście**
2 2 2	12

Wejście	**Wyjście**
3 2 4 2	45312

Szyfr Atlantydów (J)

Limit pamięci: 1024 MB Limit czasu: 1.00 s

Głęboko pod powierzchnią oceanu, wśród zatopionych ruin legendarnej Atlantydy, znajdują się starożytne tablice pokryte tajemniczymi inskrypcjami. Jasio, nurek-archeolog, musi odszyfrować układ znaków, aby odkryć pradawną wiedzę o oceanie i jego sekretach.

Jasiowi udało się już odnaleźć wszystkie N tablic z zapiskami. Na każdej z nich znajduje się N znaków. Jedyne, co pozostało naszemu bohaterowi, to ułożyć je w poprawnej kolejności. Udało mu się ustalić dwa istotne fakty:

- **W poprawnej kolejności** na i-tej tablicy znaczenie ma tylko pierwszych i znaków (pozostałe miały na celu zmylić wroga).
- Każde ułożenie tablic koduje słowo powstałe ze sklejenia **znaczących** liter z kolejnych tablic. Poprawne ułożenie koduje najmniejsze leksykograficznie takie słowo.

Przykładowo, mając dane tablice ze znakami `aab`, `aba` oraz `aca` powinniśmy je ustawić w kolejności `aca`, `aab`, `aba`, ponieważ uzyskamy w ten sposób sześcioliterowe słowo `a aa aba` (spacje pozostawiono dla czytelności), które jest leksykograficznie najmniejsze spośród wszystkich możliwych słów kodowanych przez ułożenia tablic.

Pomóż Jasiowi uporządkować starożytne inskrypcje Atlantydy!

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba całkowita N, oznaczająca zarówno liczbę tablic z inskrypcjami, jak i liczbę znaków na każdej z nich.
W kolejnych N wierszach znajduje się po jednym słowie, a każde z nich składa się z N małych liter alfabetu angielskiego.

Wyjście

Na wyjściu wypisz N wierszy, a w każdym z nich jedno spośród N-literowych słów podanych na wejściu. Słowa te powinny być wypisane w kolejności podanej w treści zadania. Jeśli istnieje wiele poprawnych odpowiedzi, wypisz dowolną z nich.

Ograniczenia

$1 \leq N \leq 5\,000$.

Przykłady

Wejście	Wyjście
3 aab aba aca	aca aab aba

Krzyżyki (K)

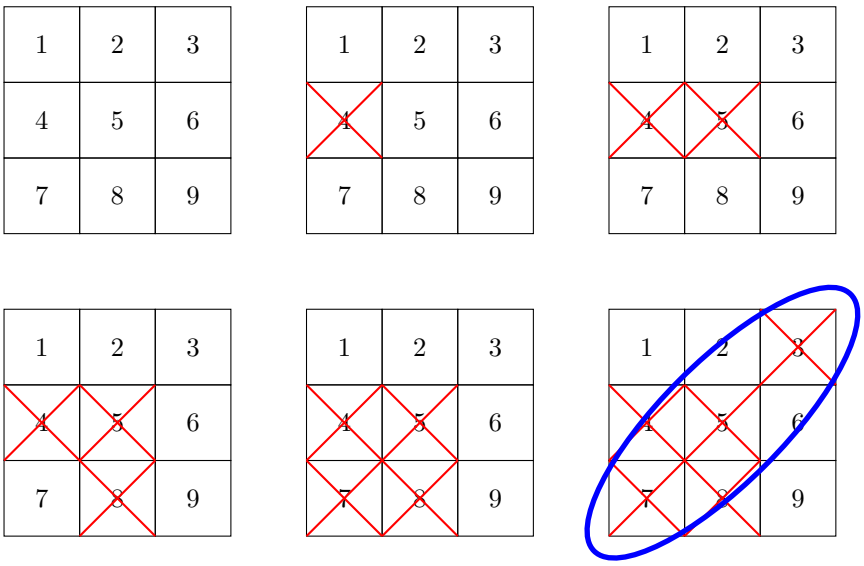
Limit pamięci: 1024 MB Limit czasu: 5.00 s

To zadanie jest interaktywne. Po wypisaniu każdego wiersza należy opróżnić bufor. W C++ możesz użyć `cout << flush`, w Pythonie – `sys.stdout.flush()`. Pamiętaj o wypisaniu białego znaku (na przykład końca linii) podczas opróżniania bufora. Należy ściśle trzymać się protokołu opisanego w rozdziale *Protokół interakcji* – niewysłanie wiadomości oczekiwanej przez interaktor może skutkować werdyktem *time limit exceeded* lub innym.

Bycie żółwiem morskim wiąże się z poważnymi nieudogodnieniami. Dla przykładu, weźmy grę w kółko i krzyżyk. Żółwie nie potrafią pisać, więc muszą sobie z nią radzić jakoś inaczej. Zaznaczanie krzyżyków jest proste – w morzu pływa masa plastikowych słomek, idealnie nadających się do tego celu. Z kółkiem jest nieco trudniej, okrągły kształt jest ciężki do odwzorowania. Należy zatem odpowiednio zmodyfikować zasady: gracze na zmianę stawiają krzyżyki na planszy 3×3. Wygrywa ten, kto pierwszy zamknie jakąś linię (wiersz, kolumnę lub przekątną). Gra ta ma bardzo proste zasady, a mimo to sprawiała żółwiom przyjemność.

Jednak w pewnym momencie, gra przestała być taka jak wcześniej, ponieważ zauważyłeś, że gracz rozpoczynający ma przewagę i jest w stanie zawsze wygrać. Naturalnie, stwierdzenie to nie zostało dobrze odebrane i większość żółwi posądziło Cię o kłamstwo. By bronić honoru, uznałeś, że zagrasz z nimi tyle gier ile tylko zechcą i wszystkie z nich wygrasz. Jeśli Ci się nie uda, zobowiązałeś się zjeść część słomek. Jak wiadomo, jedzenie słomek jest niezdrowe, dlatego lepiej tego uniknąć.[1]

Ilustracja drugiej gry z przykładowej interakcji:



[1] Byłoby to dobre miejsce, żeby przestrzec przed jedzeniem plastikowych słomek. Jednak w ostatnich czasach łatwiej znaleźć żółwia niż plastikową słomkę, więc raczej ostrzeżenie to byłoby zbędne. Żółwi też nie jedz – mają w sobie za dużo plastiku. Gdybyś się jednak zdecydował, przepis na zupę z żółwia możesz odebrać u Jury.

Protokół interakcji

Na początku należy wczytać liczbę całkowitą T, będącą liczbą gier. Następnie T razy wykonywane jest poniższe:
Rozpoczyna się gra. W każdej rundzie Twój program powinien wypisać numer pola – cyfrę od 1 do 9 (tak jak na rysunku). Jeśli powstała linia 3 krzyżyków, gra się kończy i program powinien przejść do kolejnej gry. W przeciwnym przypadku, Twój program powinien wczytać numer pola, na którym ruch robi przeciwnik. Jeśli po tym ruchu powstała linia krzyżyków, Twój program przegrał. W przeciwnym wypadku zostanie rozegrana kolejna runda gry.
Jeśli Twój program przegrał w danej grze, powinien od razu skończyć działanie, aby dostać werdykt `WA`. Inaczej, możesz otrzymać werdykt `TLE` lub inny niesprecyzowany.
Po wygraniu T-tej gry program powinien zakończyć działanie.

Ograniczenia

$1 \leq T \leq 1\,000$.

Przykładowa interakcja

Wejście	Wyjście
2	
	1
2	
	3
	4
5	
	8
7	
	3

W pierwszej grze ruchy wykonywane są kolejno na polach `1`, `2`, `3`. Te trzy pola tworzą linię. W drugiej grze ruchy wykonywane są kolejno na polach `4`, `5`, `8`, `7`, `3`. Pola `3`, `5`, `7` tworzą linię

Narzędzia do testowania lokalnego

W plikach umieszczony został program **play.py**. Skrypt ten pozwala grać ze swoim programem. Można go uruchomić za pomocą komendy:

- dla plików w C++:

  ```
  python3 play.py ./program
  ```

  gdzie `./program` jest plikiem wykonywalnym z Twoim rozwiązaniem,

- dla plików w Pythonie:

  ```
  python3 play.py program.py
  ```

Zestawy do nurkowania (L)

Limit pamięci: 1024 MB Limit czasu: 1.00 s

Jasio prowadzi wypożyczalnię sprzętu do nurkowania. Dokładniej rzecz biorąc, wypożycza on trzy elementy, które są niezbędne każdemu nurkowi: maski nurkowe, pary płetw oraz butle z tlenem.

Dzisiaj do wypożyczalni przyjechała wycieczka. Jasio wie, że każdy jej uczestnik będzie potrzebował pełnego zestawu do nurkowania, czyli maski, pary płetw oraz butli. Na stanie magazynu znajduje się aktualnie M masek, P par płetw oraz B butli z tlenem. Jednakże Jasio wie również, że dzisiaj rano wpłynęło do sklepu K zamówień na elementy sprzętu do nurkowania. Każde z nich zabiera dokładnie jeden element z wypożyczalni, lecz Jasio nie jest teraz w stanie sprawdzić który dokładnie.

Czy jesteś w stanie pomóc Jasiowi i korzystając z posiadanych przez niego informacji, obliczyć, ile co najmniej pełnych zestawów sprzętu będzie mógł wypożyczyć uczestnikom wycieczki?

Wejście

W pierwszym i jedynym wierszu wejścia podane są cztery liczby całkowite M, P, B, K, oznaczające odpowiednio, ile Jasio posiada aktualnie masek do nurkowania, par płetw i butli z tlenem oraz ile zamówień przyszło do sklepu dzisiaj rano.

Wyjście

W pierwszym i jedynym wierszu wyjścia wypisz jedną liczbę całkowitą, oznaczającą, ile co najmniej pełnych zestawów do nurkowania Jasio będzie w stanie wypożyczyć uczestnikom wycieczki.

Ograniczenia

$0 \leq M, P, B, K \leq 10^9$, $K \leq M + P + B$.

Przykłady

Wejście	Wyjście
3 4 3 1	2

Wejście	Wyjście
8 7 0 3	0

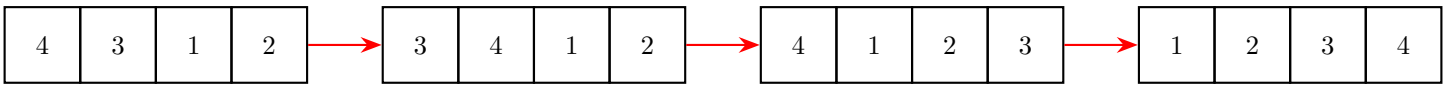
Mistrzostwa Polski Szkół Średnich w Programowaniu Zespołowym

Kraby (M)

Limit pamięci: 1024 MB Limit czasu: 2.00 s

Jasio od zawsze fascynował się życiem podwodnym. Pewnego dnia, podczas obserwacji rafy koralowej, zauważył grupę krabów wykonujących niezwykły taniec godowy. N krabów ustawiło się w szeregu, a każdy z nich miał unikalny rozmiar, który będziemy oznaczać liczbami całkowitymi od 1 do N — im większa liczba, tym większy rozmiar kraba.

Jasio doskonale wie, że taniec ten przebiega w bardzo przewidywalny sposób: dopóki pierwszym krabem od lewej nie jest najmniejszy krab, to pierwszy krab od lewej przechodzi bezpośrednio na prawo od największego kraba mniejszego od siebie, aby zaznaczyć nad nim swoją dominację. Formalnie, krab oznaczony numerem x staje po prawej stronie kraba oznaczonego numerem $x-1$.



Jako że taniec godowy mógł trwać dosyć długo (Jasio nie jest nawet do końca przekonany czy taniec ten kiedykolwiek się skończył), a tlen w butli Jasia był ograniczony, to nie doświadczył on zakończenia tańca, nad czym ubolewa.

Aby poprawić mu humor, postanowiłeś napisać program, który dla początkowej konfiguracji N krabów ustali, w jakiej konfiguracji stały pod koniec tańca godowego albo stwierdzi, że taniec ten nigdy by się nie zakończył. Czy podołasz temu zadaniu?

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba całkowita N będąca liczbą krabów uczestniczących w tańcu godowym.
W drugim wierszu wejścia znajduje się N liczb całkowitych a_i, gdzie i-ta z nich jest rozmiarem i-tego kraba.

Wyjście

Jeżeli taniec krabów nigdy się nie zakończy, to w pierwszym i jedynym wierszu wyjścia powinno się znaleźć jedno słowo `NIE`.
W przeciwnym przypadku w pierwszym wierszu wyjścia powinno się znaleźć słowo `TAK`, a w drugim wierszu wyjścia powinno znaleźć się N liczb całkowitych opisujących końcową konfigurację krabów.

Ograniczenia

$1 \leq N \leq 500\,000$, $1 \leq a_i \leq N$, $a_i \neq a_j$ dla $i \neq j$.

Przykłady

Wejście	Wyjście
4 4 3 1 2	TAK 1 2 3 4

Mistrzostwa Polski Szkół Średnich w Programowaniu Zespołowym

Rozmowy wielorybów (N)

Limit pamięci: 1024 MB Limit czasu: 1.00 s

Wieloryb Bajtoryb wraz ze swoją klasą wypłynął na szkolną wycieczkę, aby zwiedzić wrak Titanica. Podróż była długa, więc aby umilić sobie czas, uczniowie zaczęli między sobą rozmawiać.

Podczas tej wyprawy Bajtoryb dostrzegł coś niezwykle interesującego. Każdy wieloryb w jego klasie komunikował się za pomocą fal dźwiękowych określonej częstotliwości. Dokładniej, i-ty wieloryb emitował falę dźwiękową o częstotliwości a_i (która może czasami mieć wartość 0 bądź nawet ujemną). Co ciekawe, gdy dwa wieloryby używające częstotliwości a_i oraz a_j rozmawiały jednocześnie, ich fale łączyły się, tworząc nową falę. Jej częstotliwość była równa iloczynowi fal generowanych przez rozmawiające wieloryby, czyli wynosiła $a_i \cdot a_j$.

Bajtoryb zauważył, że często zdarzało się, iż nowo powstała fala przypominała falę, którą kojarzył już wcześniej z pewnym kolegą z klasy. Zaintrygowany tym zjawiskiem, zaczął się zastanawiać: czy każda możliwa rozmowa dwóch wielorybów zabrzmi jak fala jednego z jego kolegów?

Problem okazał się trudniejszy, niż początkowo sądził, więc Bajtoryb postanowił poprosić o pomoc. Czy potrafisz rozwikłać tę zagadkę i znaleźć odpowiedź?

Napisz program, który dla każdego zestawu danych wczyta częstotliwości fal generowanych przez kolejne wieloryby z klasy Bajtoryba i stwierdzi, czy każda rozmowa dwóch uczniów brzmi jak jakiś wieloryb z tej klasy.

Wejście

W pierwszym wierszu wejścia znajduje się liczba naturalna T określająca liczbę zestawów danych. W kolejnych wierszach znajdują się opisy kolejnych zestawów danych.

W pierwszym wierszu każdego zestawu danych znajduje się jedna liczba naturalna N, oznaczająca liczbę wielorybów w klasie Bajtoryba.

W kolejnym wierszu każdego zestawu danych znajduje się N liczb całkowitych a_i, pooddzielanych pojedynczymi odstępami, oznaczających częstotliwości fal generowanych przez kolejne wieloryby z klasy.

Wyjście

Na wyjściu powinno znaleźć się T wierszy.

W i-tym z nich powinna znaleźć się odpowiedź na i-ty zestaw danych, będąca jednym słowem – `TAK`, jeżeli rozmowa każdej pary wielorybów z klasy Bajtoryba brzmi jak jakiś wieloryb z tej klasy, lub `NIE` w przeciwnym wypadku.

Ograniczenia

$1 \leq T \leq 200\,000$, $1 \leq N \leq 200\,000$, $-1\,000\,000 \leq a_i \leq 1\,000\,000$.

Suma N we wszystkich zestawach danych nie przekroczy $200\,000$.

Przykłady

Wejście

```
3
3
10 0 5
5
0 -1 0 1 0
2
1 2
```

Wyjście

```
NIE
TAK
TAK
```

Wyjaśnienie

Odpowiedź dla pierwszego zestawu danych to `NIE`, ponieważ rozmowa wielorybów 1 i 3 generuje falę o częstotliwości $5 \cdot 10 = 50$. Nikt w klasie sam nie emituje takiej fali.

W drugim zestawie danych każda rozmowa dwóch wielorybów z klasy generuje falę, która jest taka sama, jak fala emitowana przez jakiegoś ucznia.

W trzecim zestawie danych jedyna możliwa rozmowa dwóch wielorybów generuje falę o częstotliwości $1 \cdot 2 = 2$, która jest taka sama jak częstotliwość fali generowanego przez drugiego ucznia.

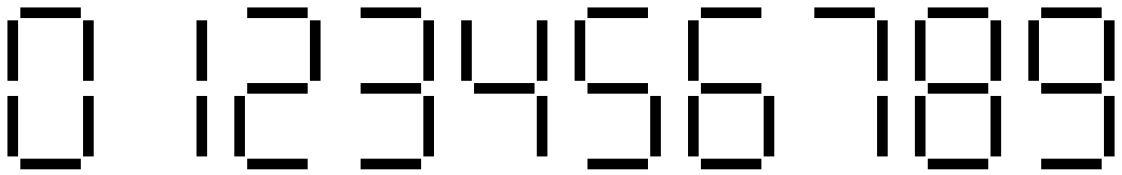
Mistrzostwa Polski Szkół Średnich w Programowaniu Zespołowym

Powrót Atlantydy (O)

Limit pamięci: 1024 MB Limit czasu: 1.00 s

W ratuszu Atlantydy znajduje się zegar, który odlicza dni do wyschnięcia wszystkich oceanów — wydarzenia, które doprowadzi do odrodzenia zatopionego miasta. Zegar składa się z $10^{10^{10}}$ wyświetlaczy siedmiosegmentowych ułożonych w jednym rzędzie, tak aby mógł wyświetlać nawet bardzo długie liczby.

Poniżej przedstawiono sposób, w jaki zegar wyświetla cyfry od 0 do 9 przy użyciu segmentów:



Pewnego dnia fan nurkowania, Jasio, przypadkiem odnalazł Atlantydę i znajdujący się w niej zegar. Zmartwiła go jednak mała liczba na wyświetlaczu — jeśli oceany rzeczywiście wyschną, nie będzie miał już gdzie nurkować! Zdecydował, że musi znaleźć sposób, aby jak najbardziej opóźnić to wydarzenie.

Na szczęście Jasio odkrył panel sterujący zegarem, który pozwala na zmianę stanu segmentów z zapalonego na zgaszony i odwrotnie. Ponieważ kończył mu się tlen, będzie mógł zmienić stan co najwyżej K segmentów.

Przed modyfikacjami Jasia, zegar wyświetlał pewną liczbę naturalną na wyświetlaczach znajdujących się najbardziej na prawo. Wszystkie użyte wyświetlacze tworzą spójny, nieprzerwany fragment, a na każdym z nich wyświetlana jest jedna z cyfr. Zegar nie wyświetla zer wiodących, więc wszystkie segmenty niepotrzebne do wyświetlenia aktualnej liczby są zgaszone. Nie zmienia to faktu, że Jasio nadal może je zapalić. Dodatkowo, jak każdy wie, Atlantyda jest jeszcze pod wodą, zatem zegar nie może na początku wyświetlać liczby 0.

Napisz program, który wczyta aktualny stan zegara i liczbę segmentów, których stan może zmienić Jasio, a następnie wypiszę największą możliwą liczbę, jaką może wyświetlać zegar po jego ingerencji.

Wejście

W pierwszym wierszu wejścia znajdują się dwie liczby naturalne N i K oddzielone pojedynczym odstępem, oznaczające odpowiednio liczbę cyfr aktualnie wyświetlanej przez zegar liczby oraz maksymalną liczbę zmian stanów segmentów, których może dokonać Jasio.
W drugim wierszu wejścia znajduje się N cyfr reprezentujących liczbę aktualnie wyświetlaną przez zegar.

Wyjście

W pierwszym i jedynym wierszu wyjścia powinna znaleźć się liczba wyświetlana przez zegar, po tym jak Jasio zmienił stany segmentów.

Ograniczenia

$1 \leq N \leq 1\,000\,000$, $2 \leq K \leq 1\,000\,000$.

Przykłady

Wejście	Wyjście
2 2 31	131