Ustawianie w pary **(pary)**

Memory limit: 64 MB Time limit: 1.00 s

Jasio i jego znajomi z klasy są przykładnymi uczniami, dlatego przed każdą lekcją ustawiają się w pary (w przypadku nieparzystej liczby osób jedna osoba zostaje bez pary).

Każdy uczeń ma swojego najlepszego przyjaciela, z którym chciałby chociaż raz być w parze. Oczywiście nikt nie jest najlepszym przyjacielem samego siebie, a ponadto każdy jest najlepszym przyjacielem dokładnie jednej osoby.

Pomóż Jasiowi wyznaczyć ile lekcji musi się odbyć, aby każdy mógł być ustawiony w parę ze swoim najlepszym przyjacielem chociaż raz.

Uwaga! Relacja przyjaźni niekoniecznie jest symetryczna, to znaczy jeśli Zbyszek jest najlepszym przyjacielem Jasia, to wcale nie oznacza, że Jasio jest najlepszym przyjacielem Zbyszka (ale może nim być).

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba naturalna N, będąca liczbą osób w klasie Jasia. W drugim wierszu znajduje się N liczb naturalnych $A_1, A_2, \ldots, A_N$, gdzie i-ta z nich oznacza indeks najlepszego przyjaciela i-tej osoby.

Wyjście

W pierwszym (jedynym) wierszu wyjścia powinna się znaleźć minimalna liczba lekcji, które muszą się odbyć, aby każdy był w parze ze swoim najlepszym przyjacielem chociaż raz.

Ograniczenia

$2 \leq N \leq 100\,000$.

Ciąg A jest permutacją, taką że $A_i \neq i$.

Przykład

Input	Output	Explanation
4 2 1 4 3	1	Wystarczy jedna lekcja, przed którą osoba numer 1 ustawi się w parę z osobą numer 2, a osoba numer 3 ustawi się w parę z osobą numer 4.

Input	Output	Explanation
3 2 3 1	3	Przed każdą lekcją w parę mogą ustawić się tylko dwie osoby, zatem potrzebne są trzy lekcje.

Dołóż kartę **(doloz-karte)**

Memory limit: 128 MB Time limit: 1.00 s

Jasio wraz z przyjaciółmi grają w swoją ulubioną grę karcianą *Dołóż kartę*.

Pewnie nigdy nie grałeś/aś w tę grę, więc pokrótce przedstawimy jej zasady. Każda z N osób siedzących w kółku ma w ręce jedną kartę, którą w swojej turze może dorzucić do stosu (jeśli to zrobi, to w ręce nie będzie już miała żadnych kart). Celem graczy jest takie dorzucanie kart do stosu, aby ich suma wyniosła K. Oczywiście chcą to zrobić w minimalnej możliwej liczbie tur. Rozgrywkę rozpoczyna gracz P-ty, potem kolej gracza $P+1$-szego itd. Po N-tym graczu następuje tura 1-szego gracza, potem 2-giego itd.

Twoim zadaniem jest wyznaczenie minimalnej potrzebnej liczby tur.

Wejście

W pierwszym wierszu wejścia znajdują się dwie liczby naturalne N oraz Q, będące odpowiednio liczbą graczy oraz liczbą gier, które rozegrają. W drugim wierszu wejścia znajduje się ciąg N liczb naturalnych $A_1, A_2, \ldots, A_N$, odpowiadających wartościom kart w rękach graczy. W i-tym z kolejnych Q wierszy znajdują się dwie liczby naturalne P_i oraz K_i, będące odpowiednio indeksem rozpoczynającego gracza oraz wymaganą do osiągnięcia sumą.

Wyjście

W i-tym wierszu należy wypisać odpowiedź dla i-tej rozgrywki.

Powinna być ona minimalną potrzebną liczbą tur, aby możliwe było osiągnięcie sumy wynoszącej K. Jeżeli jest to niemożliwe należy wypisać 0.

Ograniczenia

$1 \leq N \leq 2000$, $1 \leq Q \leq 500\,000$, $1 \leq P_i \leq N$, $1 \leq A_i, K_i \leq 5000$.

Przykład

Input	Output	Explanation
5 2 2 4 8 6 2 5 6 2 7	3 0	W pierwszej rozgrywce wystarczy, że kartę dorzuci piąty oraz drugi gracz. W drugiej rozgrywce nie jest możliwe osiągnięcie danej sumy.

Dwaj murarze **(murarze)**

Memory limit: 64 MB Time limit: 1.00 s

Jasio w drodze do szkoły natknął się na kłótnię dwóch murarzy, Bajtomira i Bajtosza. Aby rozstrzygnąć spór o to, który z nich jest lepszym majstrem, Jasio zaproponował pojedynek.

Zadaniem obu murarzy będzie wyłożenie planszy $2 \times N$ kostką brukową o wymiarach 2×1 i 2×2. Kostki 2×1 nie można obracać, musi być położona pionowo.

Jasio postanowił, że Bajtomir musi użyć parzyście wiele kostek 2×2, a Bajtosz nieparzyście wiele. Oczywiście murarze są bardzo chełpliwi, dlatego będą chcieli zaprezentować wszystkie możliwe wyłożenia planszy.

Jasiowi spieszy się do szkoły, więc to Twoim zadaniem będzie rozstrzygnięcie sporu. Jak powszechnie wiadomo murarze łatwo się nie poddają, więc poprosili Cię o pomoc w wielu pojedynkach.

Napisz program stwierdzający, dla każdego z pojedynków, który z murarzy jest w stanie zaprezentować więcej możliwości wyłożenia planszy.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba naturalna Z, będąca liczbą pojedynków do rozstrzygnięcia.

W i-tym z kolejnych Z wierszy znajduje się jedna liczba naturalna N_i, będąca szerokością planszy w i-tym pojedynku.

Wyjście

W i-tym wierszu należy wypisać rezultat i-tego pojedynku.

Powinien być on jednym z napisów `Bajtomir`, `Bajtosz` lub `Remis`, w zależności od tego, który z murarzy jest w stanie zaprezentować więcej możliwości wyłożenia planszy.

Ograniczenia

$1 \leq Z \leq 100\,000$, $1 \leq N_i \leq 10^{18}$.

Przykład

Input	Output	Explanation
2 2 4	Remis Bajtosz	Dla $N = 2$ istnieje jedno kafelkowanie Bajtomira i jedno kafelkowanie Bajtosza. Dla $N = 4$ istnieją dwa kafelkowania Bajtomira i trzy kafelkowania Bajtosza.

Skoczek (skoczek)

Memory limit: 64 MB Time limit: 1.00 s

Jasio dostał na urodziny nową grę, której plansza składa się z N kolejno ułożonych pól. Na każdym z pól znajduje się pewna liczba kłujących kolców.

Przed pierwszym polem stoi skoczek, którego zadaniem jest dostanie się za ostatnie pole planszy, wbijając sobie przy tym jak najmniej kolców. W każdym kroku skoczek może przemieścić się na następne pole planszy albo wykonać skok o **co najwyżej** D pól do przodu. Zgodnie z instrukcją, taki skok może zostać wykonany **co najwyżej trzy** razy.

Twoim zadaniem jest policzenie minimalnej długości skoku D, takiej że skoczek jest w stanie przejść po planszy, wbijając sobie przy tym **co najwyżej** K kolców.

Wejście

W pierwszym wierszu wejścia znajdują się dwie liczby naturalne N oraz K, będące odpowiednio rozmiarem planszy oraz ograniczeniem na liczbę kolców. W drugim wierszu znajduje się N liczb naturalnych $A_1, A_2, \ldots, A_N$, gdzie i-ta z nich oznacza liczbę kolców na i-tym polu.

Wyjście

W pierwszym (jedynym) wierszu wyjścia powinna się znaleźć minimalna (dodatnia) długość skoku D, taka że skoczek jest w stanie przejść po planszy, zbierając przy tym co najwyżej K kolców.

Ograniczenia

$1 \leq N \leq 100\,000$, $1 \leq K, A_i \leq 10^9$.

Przykład

Input	Output	Explanation
10 5 3 1 1 4 5 2 1 3 5 1	3	Skoczek może kolejno: skoczyć na trzecie pole, skoczyć na szóste pole, przejść na siódme pole, skoczyć na dziesiąte pole i przejść na pole poza planszą.

Input	Output	Explanation
5 2 3 4 5 6 7	6	Skoczek musi od razu przeskoczyć całą planszę.

LCS i podsłowo **(lcs-podslowo)**

Memory limit: 256 MB Time limit: 1.00 s

Po kursie z algorytmów tekstowych, Jasio nadal ma problem z odróżnieniem podciągu od podsłowa. W związku z tym, prowadzący postanowił przygotować dla niego specjalne zadanie. Dla danych słów A, B oraz S, Jasio ma obliczyć długość najdłuższego wspólnego podciągu słów A i B, takiego że zawiera S jako podsłowo.

W ramach przypomnienia: podciąg to słowo, powstałe poprzez usunięcie pewnych (być może żadnych) liter ze słowa, a następnie odczytanie pozostałych liter od lewej do prawej. Natomiast podsłowo to słowo, powstałe poprzez usunięcie pewnych (być może żadnych) liter z początku oraz końca słowa, a następnie odczytanie pozostałych liter od lewej do prawej. Na przykład dla słowa *alekoala*, jednym z jego podciągów jest *eol*, a jednym z jego podsłów jest *lek*.

Wejście

W pierwszych trzech wierszach wejścia znajdują się trzy słowa (składające się z małych liter alfabetu angielskiego) A, B oraz S.

Wyjście

W pierwszym (jedynym) wierszu wyjścia powinna się znaleźć długość najdłuższego wspólnego podciągu słów A i B, takiego że zawiera S jako podsłowo. Jeżeli taki podciąg nie istnieje, Twój program powinien wypisać 0.

Ograniczenia

Długość każdego z napisów na wejściu nie przekracza 5000 znaków.

Przykład

Input	Output
alekoala alxekkoxa lek	6

Liczby dwucyfrowe **(liczby-dwucyfrowe)**

Memory limit: 64 MB Time limit: 1.00 s

Jak pewnie wiesz, Jasio jest fanatykiem liczb, a szczególnie interesują go takie nieujemne liczby całkowite, których zapis dziesiętny składa się z co najwyżej dwóch różnych cyfr: A oraz B.

Twoim zadaniem jest obliczenie, ile jest takich liczb nie większych od N.

Wejście

W pierwszym (jedynym) wierszu wejścia znajdują się trzy liczby naturalne N, A i B.

Wyjście

W pierwszym (jedynym) wierszu wyjścia powinna się znaleźć liczba nieujemnych liczb całkowitych nie większych od N, których zapis dziesiętny składa się z co najwyżej dwóch różnych cyfr: A oraz B.

Ograniczenia

$1 \leq N \leq 10^{18}$, $0 \leq A < B \leq 9$.

Przykład

Input	Output	Explanation
`100 1 3`	`6`	Te liczby to: $1, 3, 11, 13, 31, 33$.

Podział łupów (podzial-lupow)

Memory limit: 64 MB Time limit: 1.00 s

Praprapradziadek Jasia zwany Jasiomirem trudnił się poszukiwaniem skarbów. W jego wyprawach towarzyszyli mu dwaj wierni kompani Wojemił oraz Miłorad. Po jednej ze swoich wypraw bohaterowie postanowili równo podzielić się wszystkimi zdobytymi złotymi monetami.

Każdy z trójki kompanów niesie pewną liczbę monet w swoim plecaku, a ich zadaniem jest takie przełożenie tych monet, aby wszyscy mieli ich tyle samo (w swoich plecakach). Oczywiście, aby wyciągnąć lub włożyć monety do plecaka, musi on zostać otworzony. Niestety, plecaki posiadają najwyższej jakości zabezpieczenia, których obejście zajmuje dość długo, dlatego bohaterom zależy na zminimalizowaniu liczby otwieranych plecaków.

Wejście

W pierwszym (jedynym) wierszu wejścia znajdują się trzy liczby naturalne A, B i C, odpowiadające liczbie monet w plecakach Jasiomira, Wojemiła oraz Miłorada.

Wyjście

W pierwszym (jedynym) wierszu wyjścia powinna się znaleźć jedna liczba naturalna, będąca minimalną liczbą plecaków koniecznych do otwarcia, aby możliwe było dokonanie równego podziału wszystkich monet lub słowo `NIE`, jeśli dokonanie takiego podziału nie jest możliwe.

Ograniczenia

$1 \leq A, B, C \leq 10^8$.

Przykład

Input	Output	Explanation
`1 4 4`	`3`	Wszystkie plecaki muszą zostać otworzone, aby możliwe było dokonanie równego podziału monet.

Input	Output	Explanation
`3 3 4`	`NIE`	Dokonanie równego podziału monet nie jest możliwe.

Digimony i pokemony **(digimony)**

Memory limit: 64 MB Time limit: 2.00 s

Jasio jest zapalonym kolekcjonerem digimonów, pokemonów i innych stworów. Niektóre z nich chciałby dobrać w pary i poukładać na półeczce, a resztę schować do skrzyni. Oczywiście dobieranie w pary nie może być byle jakie, każdy stwór w różnym stopniu lubi (albo nie) każdego innego stwora.

Na potrzeby zadania założymy, że Jasio posiada N digimonów i M pokemonów. Ponadto przygotował on Q potencjalnych par stworów, wraz z ich preferencjami. Jasio może dobrać stwory zgodnie z przygotowanymi parami, jednak musi zadbać o to, aby wszystkie stwory były zadowolone.

Dla danego dobrania w pary, pewna niewybrana para stworów (znajdująca się na liście potencjalnych par) będzie niezadowolona, jeśli zajdą oba poniższe warunki:

- digimon z tej pary nie ma przyporządkowanego partnera lub bardziej lubi pokemona z tej pary, niż tego z którym obecnie jest dobrany,
- pokemon z tej pary nie ma przyporządkowanego partnera lub bardziej lubi digimona z tej pary, niż tego z którym obecnie jest dobrany.

Twoim zadaniem jest wyznaczenie minimalnej i maksymalnej liczby możliwych par.

Wejście

W pierwszym wierszu wejścia znajdują się trzy liczby naturalne N, M oraz Q, będące odpowiednio liczbą digimonów, pokemonów oraz potencjalnych par. W i-tym z kolejnych Q wierszy znajdują się cztery liczby naturalne D_i, P_i, A_i oraz B_i, będące odpowiednio numerem digimona, numerem pokemona, preferencją digimona (względem pokemona) oraz preferencją pokemona (względem digimona). Im wyższa preferencja tym bardziej dany stwór lubi tego drugiego. Digimony ponumerowane są kolejnymi liczbami naturalnymi od 1 do N, a pokemony ponumerowane są kolejnymi liczbami naturalnymi od 1 do M.

Wyjście

W pierwszym (jedynym) wierszu wyjścia powinny się znaleźć dwie liczby naturalne, będące odpowiednio minimalną i maksymalną liczbą możliwych par. Jeżeli nie istnieje żadne poprawne dobranie w pary, to należy wypisać `NIE`.

Ograniczenia

$1 \leq N, M \leq 100\,000$, $1 \leq Q \leq 500\,000$, $1 \leq D_i \leq N$, $1 \leq P_i \leq M$.

Zarówno ciąg A jak i ciąg B tworzą permutację.

Każda możliwa para stworów występuje co najwyżej raz.

Przykład

Input	Output	Explanation
3 2 3 1 2 3 2 2 1 1 3 3 2 2 1	2 2	Jedynym możliwym sposobem jest dobranie w parę pierwszego digimona z drugim pokemonem oraz drugiego digimona z pierwszym pokemonem.

Fibonacci z utrudnieniem **(fibonacci-hard)**

Memory limit: 64 MB Time limit: 3.00 s

Jasio postanowił pomóc w przerzucaniu zadań na nowy system Solve 4. W tym celu wybrał jedno z zadań znajdujących się w Solvie 3 i nieznacznie je zmodyfikował. Niestety, tak zmodyfikowane zadanie okazało się dla Jasia zbyt trudne, więc poprosił Cię o pomoc.

Twoim zadaniem jest obliczenie wartości $Fib(Fib(Fib(N)))$, gdzie $Fib(N)$ oznacza N-tą liczbę Fibonacciego.

W ramach przypomnienia, ciąg liczb Fibonacciego definiujemy zgodnie ze wzorem: $Fib(0) = 0$, $Fib(1) = 1$, $Fib(N + 2) = Fib(N + 1) + Fib(N)$ dla $N \geq 0$.

Jako że wynik może okazać się zbyt duży do pomieszczenia w znanym wszechświecie, wystarczy że wypiszesz jego resztę z dzielenia przez $998\,244\,353$.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba naturalna Z, będąca liczbą zestawów testowych.

W i-tym z kolejnych Z wierszy znajduje się jedna liczba naturalna N_i, będąca wartością, dla której należy obliczyć wynik.

Wyjście

W i-tym wierszu należy wypisać odpowiedź dla i-tego zestawu testowego.

Powinna być ona resztą z dzielenia przez $998\,244\,353$ wartości $Fib(Fib(Fib(N_i)))$.

Ograniczenia

$1 \leq Z \leq 100\,000$, $1 \leq N_i \leq 10^{18}$.

Przykład

Input	Output	Explanation
2 2 6	1 10946	Dla $N = 2$ mamy $Fib(2) = 1$ i $Fib(1) = 1$, zatem $Fib(Fib(Fib(2))) = 1$. Dla $N = 6$ mamy $Fib(6) = 8$, $Fib(8) = 21$ i $Fib(21) = 10946$, zatem $Fib(Fib(Fib(6))) = 10946$.