Intensywny trening (A)

Limit pamięci: 512 MB Limit czasu: 2.00 s

Janek bardzo lubi brać udział w internetowych konkursach algorytmicznych. Przez ostatnie N dni trenował, biorąc udział w konkursach na dwóch swoich ulubionych platformach: Mouseforces i Ratcoder. Każdego dnia zapisywał swój ranking będący liczbą całkowitą. Ranking i-tego dnia na platformie Mouseforces oznaczamy przez X_i, a na platformie Ratcoder Y_i. Po N dniach treningu chciałby sprawdzić swoje wyniki. W tym celu z każdego dnia wypisze jeden z rankingów X_i lub Y_i i zapisze kolejno na kartce. Ponieważ Janek bardzo lubi oglądać swój progres, chce, żeby ciąg rankingów wypisany na kartce był rosnący. Janek jest zmęczony po bardzo intensywnym treningu, więc poprosił cię o pomoc. Twoim zadaniem jest skonstruować taki ciąg lub stwierdzić, że jest to niemożliwe.

Wejście

W pierwszym wierszu znajduje się jedna liczba N, opisująca liczbę dni, podczas których Janek brał udział w konkursach. W kolejnych N wierszach znajdują się liczby X_i i Y_i oddzielone spacją, opisujące rankingi i-tego dnia.

Wyjście

W pierwszym wierszu wypisz "Tak", jeśli można z każdej pary wybrać jedną liczbę tak, żeby utworzyć ciąg rosnący, i "Nie" w przeciwnym razie. Jeśli odpowiedź w pierwszym wierszu to "Tak", to w drugim wierszu wypisz N liczb $R_1, R_2, \ldots, R_N$ takich, że $R_i = X_i$ lub $R_i = Y_i$, oraz ciąg R jest rosnący. Jeśli istnieje więcej niż jeden poprawny ciąg, należy wypisać dowolny.

Ograniczenia

$1 \leq N \leq 5 \cdot 10^5$
$1 \leq X_i \leq 10^9$
$1 \leq Y_i \leq 10^9$

Przykład

Wejście	Wyjście
3 2 3 5 3 4 4	Tak 2 3 4

Wejście	Wyjście
3 1 2 8 3 2 3	Nie

Pilates (B)

Limit pamięci: 128 MB Limit czasu: 0.50 s

Janek, oprócz uczestnictwa w konkursach programistycznych, dba o swoje zdrowie, uprawiając pilates. Jednym z elementów jego treningu jest wykonanie określonej przez komputer liczby pompek a oraz przysiadów b. Janek, będąc sprytnym hakerem, zauważył, że może oszukać program, pod warunkiem, że liczba pompek c będzie miała tę samą resztę z dzielenia przez 2 co a, a liczba brzuszków d będzie miała tę samą resztę z dzielenia przez 3 co b, przy czym suma $a + b$ będzie miała tę samą resztę z dzielenia przez 5 co suma $c + d$. Janek nie lubi się przemęczać, dlatego chciałby, aby suma $c + d$ była jak najmniejsza. Janek jednocześnie chce, choć odrobinę zadbać o formę więc planuje zawsze zrobić co najmniej jedną pompkę i przysiad. Janek nie jest pewny, jakie wartości a i b dostanie na kolejnym treningu, dlatego chciałby przygotować się na t różnych przypadków.

Wejście

W pierwszej linii znajduje się liczba t — liczba przypadków testowych.
W kolejnych t liniach znajdują się dwie liczby całkowite a oraz b.

Wyjście

Dla każdego przypadku testowego program powinien wypisać najmniejszą możliwą sumę $s = c + d$, dla której istnieją dodatnie liczby c i d, spełniające warunki zadania.

Ograniczenia

$1 \leq t \leq 1000$
$1 \leq a, b \leq 10^9$

Przykład

Wejście	Wyjście	Wyjaśnienie
3 5 7 12 6 1 3	2 8 4	Dla pierwszego przypadku testowego Janek może wykonać jedną pompkę oraz jeden przysiad czym zadowoli program. W drugim przypadku Janek może wykonać 2 pompki oraz 6 przysiadów. W trzecim przypadku nie może ułatwić treningu.

Prezent (C)

Limit pamięci: 512 MB Limit czasu: 2.00 s

Janek, będąc bardzo zajęty treningiem do olimpiady, nie zauważył, że święta już dawno minęły. Pomimo to zdecydował się napisać jeszcze list do Świętego Mikołaja z prośbą o zaległy prezent. Janek, podobnie jak rok wcześniej, poprosi o permutację n liczb, jednak chce, żeby była jak najmniej podobna do tej, którą dostał rok wcześniej. W tym celu zdefiniował współczynnik podobieństwa dwóch permutacji p i q jako $\sum_i nwd(p_i, q_i)$, gdzie $nwd(x, y)$ oznacza największy wspólny dzielnik liczb x i y. Janek chce poprosić o taką permutację q, która zminimalizuje współczynnik podobieństwa z permutacją p, którą dostał w tamtym roku. Ponieważ Janek jest bardzo zajęty implementowaniem zadania z geometrii na następne kółko, prosi Cię o pomoc w znalezieniu dowolnej takiej permutacji.

Wejście

W pierwszym wierszu znajduje się n, długość permutacji.
W drugim wierszu n liczb $p_1, p_2, \ldots, p_n$, oznaczających permutację p.

Wyjście

W pierwszym i jedynym wierszu wypisz n różnych liczb $q_1, q_2, \ldots, q_n$, czyli permutację q mającą minimalny współczynnik podobieństwa z p. Jeśli jest więcej niż jedna taka permutacja, możesz wypisać dowolną.

Ograniczenia

$1 \leq n_i \leq 2 \cdot 10^5$
$1 \leq p_i \leq n$, $p_i \neq p_j$ dla $1 \leq i \neq j \leq n$

Przykład

Wejście	Wyjście
3 2 1 3	1 3 2

Wejście	Wyjście
7 3 1 7 4 2 5 6	4 3 6 7 5 2 1

Racer (D)

Limit pamięci: 128 MB Limit czasu: 2.00 s

Janek po zmaksowaniu finału Olimpiady Informatycznej w połowie czasu chce zabić czas przy swojej ulubionej grze Math Racer. Niestety, Janek nie jest najlepszy z matematyki, więc w ustawieniach wyłączył obliczeniową część rozgrywki.

W nowej wersji Janek znajduje się w $1.$ wierszu planszy o wymiarach 3×10^9. Może poruszać się na dwa sposoby:
O jedno pole w dół w tej samej kolumnie.
Po skosie w dół w lewo lub w prawo (o ile nie wychodzi poza planszę).

Ponadto, na n parami różnych pozycjach znajdują się blokady, które nie pozwalają wjechać na część pól, np. `4 xx.` oznacza, że w $4.$ wierszu zablokowane są pola w $1.$ i $2.$ kolumnie. Zapory nie znajdują się w pierwszym ani w ostatnim wierszu.

Wejście

W pierwszym wierszu wejścia znajduje się n – liczba barier.
Następnie n linii, z których każda zawiera:
Liczbę a_i – numer wiersza, w którym znajduje się blokada.
Ciąg trzech znaków, składający się z "x" i "."
x oznacza pole zablokowane.
. oznacza pole dostępne.
Liczby a_i są parami różne.

Wyjście

Wypisz TAK, jeśli możliwe jest dotarcie do wiersza 10^9, omijając bariery.
Wypisz NIE, jeśli to niemożliwe.
Pamiętaj, aby wypisać TAK/NIE dużymi literami.

Ograniczenia

$1 \leq n \leq 100\,000$ – liczba barier.
$2 \leq a_i \leq 10^9 - 1$ – numery wierszy, w których znajdują się bariery.

Przykład

Wejście	Wyjście
3 5 x.. 3 x.x 4 .xx	TAK

Wejście	Wyjście
2 12345678 xx. 12345679 .xx	NIE

Wejście	Wyjście
1 5 xxx	NIE

Zgłoszenia (E)

Limit pamięci: 512 MB Limit czasu: 2.00 s

Janek dużo trenuje, aby wysokimi wynikami na konkursach programistycznych imponować znajomym dziewczynom. Na ostatnim konkursie napotkał nietypowe zadanie, które składa się z n podzadań. Za i-te podzadanie można otrzymać a_i punktów. Po wysłaniu zgłoszenia Janek otrzymuje informację o łącznej liczbie punktów, które uzyskał, ale nie dowiaduje się, które podzadania zostały zaliczone.

Janek chce odpowiedzieć na q zapytań. Każde zapytanie składa się z dwóch liczb całkowitych x oraz y, które oznaczają liczbę punktów uzyskanych przez niego w dwóch zgłoszeniach. Dla każdego zapytania Janek chciałby wiedzieć, ile maksymalnie punktów można uzyskać, uwzględniając oba zgłoszenia, przy założeniu, że każde podzadanie może zostać zaliczone tylko raz. Jeśli nie da się uzyskać takiej liczby punktów, należy wypisać -1.

Wejście

Na wejściu znajdują się:
Dwie liczby całkowite n oraz q — liczba podzadań oraz liczba zapytań.
n liczb całkowitych $a_1, a_2, \ldots, a_n$, gdzie a_i oznacza liczbę punktów za i-te podzadanie.
q par liczb całkowitych x oraz y — wyniki dwóch zgłoszeń dla każdego zapytania.

Wyjście

Dla każdego z q zapytań wypisz jedną liczbę: Maksymalną liczbę punktów, które można uzyskać za dane zadanie, biorąc pod uwagę oba zgłoszenia.
Jeśli nie da się uzyskać dokładnie takiej liczby punktów, wypisz -1.

Ograniczenia

$1 \leq n \leq 100$, $1 \leq q \leq 5000$
$\sum_{i=1}^{n} a_i \leq 1000$
$0 \leq x, y \leq \sum_{i=1}^{n} a_i$

Przykład

Wejście	Wyjście	Wyjaśnienie
3 3 3 4 2 5 4 3 3 1 4	9 3 -1	W pierwszym zapytaniu drugie zgłoszenie mogło zaliczyć podzadanie 1. oraz 3. a drugie podzadanie 2. W drugim oba zgłoszenia musiały zaliczyć podzadanie 1. W trzecim zapytaniu nie ma możliwości, żeby zgłoszenie dostało 1 punkt.

Gra (F)

Limit pamięci: 256 MB Limit czasu: 2.00 s

Janek gra ze swoim przyjacielem Karolem w grę. Na stole znajduje się n kubków, ponumerowanych kolejnymi liczbami naturalnymi od 1 do n, z których każdy ma przypisaną wartość. Oprócz kubków dostępny jest multizbiór przedziałów, a każdy z graczy ma nieograniczoną liczbę kulek. Gra przebiega w systemie turowym, w którym gracze wykonują ruchy naprzemiennie, a zaczynającym jest Janek.

- W każdej turze gracz wybiera jeden przedział (oznaczmy jako $[l, r]$) z multizbioru i wrzuca po jednej kulce do każdego kubka o numerze z przedziału $[l, r]$.
- Po wrzuceniu kulek do kubków wybrany przedział zostaje usunięty z dostępnego multizbioru.
- Janek zawsze zaczyna grę.
- Po zakończeniu gry wynikiem Janka jest maksymalna wartość kubka, w których ma więcej kulek niż Karol lub -1, jeśli nie ma żadnego takiego kubka.
- Janek stara się maksymalizować swoją wartość, podczas gdy Karol dąży do jej zminimalizowania.
- Gra kończy się w momencie, kiedy żaden z graczy nie może wykonać ruchu, ponieważ wszystkie przedziały zostały wykorzystane.

Twoim zadaniem jest określenie, jaki maksymalny wynik może osiągnąć Janek, biorąc pod uwagę optymalną strategię obu graczy.

Wejście

Pierwszy wiersz wejścia zawiera liczbę t - przypadków testowych. Następnie dla każdego przypadku testowego:

- Liczba całkowita n — liczba kubków.
- n liczb całkowitych $a_1, a_2, \ldots, a_n$ — wartości kubków.
- Liczba całkowita m — liczba dostępnych przedziałów.
- m par liczb całkowitych l_i, r_i — przedziały, do których można wrzucać kulki.

Wyjście

Dla każdego przypadku testowego program powinien wypisać maksymalny wynik, który może uzyskać Janek, uwzględniając optymalną strategię obu graczy.

Ograniczenia

- $t \leq 10\,000$ — liczba przypadków testowych.
- $1 \leq n \leq 100\,000$.
- $1 \leq a_i \leq 10^9$.
- $1 \leq m \leq 100\,000$.
- $1 \leq l_i \leq r_i \leq n$.
- Suma wartości n oraz suma wartości m we wszystkich przypadkach testowych nie przekroczą $2 \cdot 10^5$.

Przykład

Wejście	Wyjście	Wyjaśnienie

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

```
-1
3
```

Dla pierwszego przypadku testowego Janek musi wybrać przedział 1 1, taki sam przedział wybiera Karol i w żadnym kubku Janek nie ma większej liczby kulek. W drugim przypadku testowym Janek może wybrać przedział 2 4, Karol będzie musiał wybrać przedział 1 3. Wynikiem Janka jest liczba 3. Jest to wartość kubka numer 4, w którym Janek ma jedną kulkę, a Karol nie ma żadnej kulki.

Finał (G)

Limit pamięci: 512 MB Limit czasu: 2.00 s

Janek podczas swojego startu w finale olimpiady informatycznej napotkał następujące zadanie.
Dane jest drzewo o n wierzchołkach. Każdy wierzchołek ma swoją wagę a_i. Należy znaleźć największą wartość $\sum_i i \cdot a_{p_i}$, gdzie ciąg $p_1, p_2, \ldots, p_n$ oznacza dowolną kolejność odwiedzania wierzchołków przez algorytm DFS.
Rozważmy następujący pseudokod:

```
p = []
odwiedzony = [0,0,...,0]
void dfs(v):
    dodaj v na koniec p
    weź dowolną permutację s wierzchołków u takich, że istnieje krawędź (u,v) oraz odwiedzony
    dla wszystkich u należących do s:
        dfs(u)
```

Formalnie $p_1, p_2, \ldots, p_n$ jest permutacją, która może powstać poprzez wywołanie funkcji $dfs(s)$ dla dowolnego wierzchołka startowego s.

Jak już wiesz z któregoś z poprzednich zadań, Janek rozwiązał to zadanie z olimpiady bez problemu, czy Ty też dasz radę?

Wejście

W pierwszym wierszu znajduje się T, liczba przypadków testowych.
Dla każdego przypadku testowego:
W pierwszym wierszu liczba całkowita n.
W drugim wierszu n dodatnich liczb całkowitych $a_1, a_2, \ldots, a_n$, oznaczających wagi wierzchołków.
Potem w kolejnych $n-1$ wierszach dwie liczby u i v opisujące krawędź między dwoma wierzchołkami w drzewie.
Graf na wejściu tworzy drzewo.

Wyjście

T wierszy, w każdym jedna liczba, odpowiedź do danego testu.

Ograniczenia

Liczba przypadków testowych spełnia $1 \leq T \leq 10^4$
$1 \leq n_i$, $\sum_i n_i \leq 2 \cdot 10^5$
$1 \leq a_i \leq 10^6$
$1 \leq u \neq v \leq n_i$

Przykład

Wejście	Wyjście	Wyjaśnienie

```
2
6
4 1 9 2 2 4
3 5
1 2
5 1
1 6
2 4
8
3 4 6 7 1 3 2 3
1 2
2 3
3 4
4 5
5 6
6 7
7 8
```

```
97
148
```

Dla pierwszego przypadku testowego poprawną kolejnością DFS dającą maksymalny wynik jest $(2, 4, 1, 6, 5, 3)$.