Dynamiczna permutacja **(dynamiczna-permutacja)**

Limit pamięci: 64 MB Limit czasu: 5.00 s

W tym zadaniu celem jest przygotowanie algorytmu/struktury danych do zarządzania dynamiczną permutacją: pewnym ustawieniem liczb $1, 2, \ldots, N$ w ciąg.

Na początku otrzymujemy ciąg π: ciąg parami różnych liczb $\pi[1], \pi[2], \ldots, \pi[N]$ z przedziału 1 do N. Następnie, należy obsłużyć operacje/zapytania następujących typów:

- zamień $\pi[i]$ oraz $\pi[j]$,
- wyznacz największą wartość spośród $z, \pi[z], \pi[\pi[z]], \pi[\pi[\pi[z]]], \ldots, \pi^k[z]$.

Napisz program, który wczyta początkową permutację oraz operacje i zapytania, wyznaczy odpowiedzi na wszystkie zapytania i wypisze wyniki na standardowe wyjście.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba naturalna N, oznaczająca długość permutacji. W drugim wierszu wejścia znajduje się N parami różnych liczb naturalnych π_i, pooddzielanych pojedynczymi odstępami. W trzecim wierszu wejścia znajduje się jedna liczba naturalna Q, oznaczająca liczbę operacji/zapytań. W kolejnych Q wierszach znajdują się operacje/zapytania, po jednym w wierszu. Każde zapytanie jest postaci:

- `swap` x_i y_i – dla wartości $1 \leq x_i, y_i \leq N$, wykonaj zamianę $\pi[x_i]$ oraz $\pi[y_i]$,
- `query` z_i k_i – dla wartości $1 \leq z_i \leq N$, podaj największą wartość ze zbioru $\{z_i, \pi[z_i], \pi[\pi[z_i]], \ldots, \pi^{k_i}[z_i]\}$.

Wyjście

Dla każdego zapytania `query`, zgodnie z kolejnością na wejściu należy wypisać odpowiedź. Odpowiedzi dla zapytań należy wypisywać w osobnych wierszach, bez dodatkowych odstępów.

Ograniczenia

$1 \leq N \leq 100\,000$, $1 \leq Q \leq 100\,000$, $1 \leq k_i \leq N$.

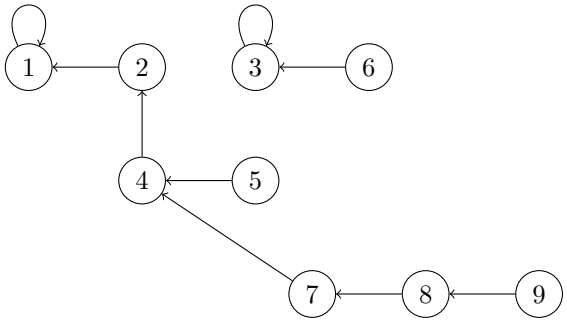
Przykład

Wejście	Wyjście
5 2 3 1 5 4 7 query 2 3 query 2 1 swap 3 5 query 2 3 query 1 5 swap 5 1 query 2 1	3 3 5 5 3

Problem grafowy **(problem-grafowy)**

Limit pamięci: 32 MB Limit czasu: 1.00 s

Jasio przygotowuje się do Bajtockiej Olimpiady Informatycznej Juniorów. Wie, że jego słabą stroną obecnie są grafy. Dlatego trenuje zadania grafowe bardzo intensywnie. Ostatnio natknął się na problem o grafach funkcyjnych: grafach, w których każdy wierzchołek ma dokładnie jedną krawędź wychodzącą. Po takim grafie łatwo nawigować, wystarczy w każdym kroku podążyć krawędzią (jedyną) w stronę następnego wierzchołka.



W zadaniu, z którym walczy teraz Jasio, trzeba dla każdego wierzchołka i wyznaczyć sumę numerów wierzchołków, na jakie natkniemy w krokach o parzystych numerach wykonując taki spacer i kończąc go w wierzchołku, którego krawędź prowadzi do samego siebie. Jasio nie wie czy to coś zmienia, ale zauważył, że we wszystkich testach w tym zadaniu dla każdego wierzchołka i jego krawędź prowadzi do wierzchołka o numerze mniejszym niż i lub równym i.

Pomóż Jasiowi rozwiązać zadanie i trenować dalej.

Napisz program, który: wczyta opis grafu, wyznaczy dla każdego wierzchołka i sumę numerów wierzchołków na odległościach parzystych od i i wypisze wyniki na standardowe wyjście.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba naturalna N, określająca liczbę wierzchołków grafu. W drugim wierszu wejścia znajduje się ciąg N liczb naturalnych T_i, pooddzielanych pojedynczymi odstępami – i-ta liczba określa, że krawędź z wierzchołka numer i wychodzi do wierzchołka numer T_i.

Wyjście

Twój program powinien wypisać na wyjście dokładnie N wierszy. W i-tym wierszu powinna się znaleźć odpowiedź dla wierzchołka i.

Ograniczenia

$1 \leq N \leq 500\,000$, $1 \leq T_i \leq i$.

Przykład

Wejście

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

Wyjście

```
1
2
3
5
7
6
9
13
18
```

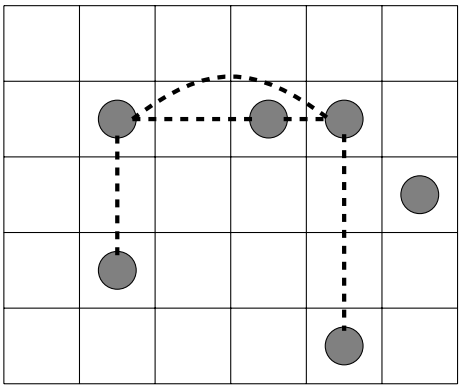
Wyjaśnienie

Ostatnią liczbą na wyjściu ma być $9 + 7 + 2 = 18$, ponieważ wierzchołki ze zbioru $\{9, 7, 2\}$ są w parzystych odległościach od wierzchołka 9.

Widoczność **(widocznosc)**

Limit pamięci: 64 MB Limit czasu: 2.00 s

Na niektórych polach prostokątnej planszy znajdują się roboty. Każdy robot ma umieszczone cztery kamery pozwalające mu zobaczyć każdego innego robota (o ile takie są) w jednym z czterech podstawowych kierunków (góra, dół, lewo, prawo), w tym samym wierszu lub tej samej kolumnie, w której ów robot się znajduje.



Napisz program, który: wczyta pozycje robotów, wyznaczy liczbę par robotów, które widzą się nawzajem i wypisze wynik na standardowe wyjście.

Wejście

W pierwszym wierszu wejścia znajduje się jedna liczba naturalna N, określająca liczbę robotów. W kolejnych N wierszach znajdują się pary liczb x_i oraz y_i, oddzielone pojedynczym odstępem. Są to współrzędne pola, na którym znajduje się i-ty robot, odpowiednio numer wiersza oraz numer kolumny.

Wiersze i kolumny numerowane są kolejnymi liczbami naturalnymi.

Gwarantowane jest, że pozycje wszysktich robotów są parami różne.

Wyjście

W pierwszym (jedynym) wierszu wyjścia powinna się znaleźć jedna liczba całkowita – liczba (nieuporządkowanych) par robotów, które się widzą nawzajem.

Ograniczenia

$1 \leq N \leq 500\,000$, $1 \leq x_i, y_i \leq 10^9$.

Przykład

Wejście	Wyjście	Wyjaśnienie
6 5 5 2 2 2 4 4 2 2 5 3 6	5	Sytuację z tego testu przykładowego obrazuje rysunek w treści powyżej.