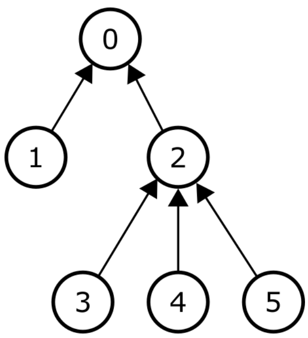
Baum

Betrachte einen **Baum** aus N **Knoten**, nummeriert von 0 bis $N-1$. Knoten 0 heisst **Wurzel**. Jeder Knoten mit Ausnahme der Wurzel hat genau einen Elternknoten. Für jedes i mit $1 \leq i < N$ gilt, dass der Elternknoten von i der Knoten $P[i]$ ist, wobei $P[i] < i$. Wir nehmen auch an, dass $P[0] = -1$.

Für jeden Knoten i ($0 \leq i < N$) ist der **Teilbaum** von i die Menge der folgenden Knoten:

- i, und
- jeder Knoten, dessen Elternknoten i ist, und
- jeder Knoten, dessen Grosselternknoten i ist, und
- jeder Knoten, dessen Urgrosselternknoten i ist,
- etc.

Das untenstehende Bild zeigt einen Beispielbaum aus $N = 6$ Knoten. Jeder Pfeil verbindet einen Knoten mit seinem Elternknoten, bis auf die Wurzel, die keinen Elternknoten hat. Der Teilbaum von Knoten 2 enthält Knoten $2, 3, 4$ und 5. Der Teilbaum von Knoten 0 enthält alle 6 Knoten des Baumes und der Teilbaum von Knoten 4 enthält nur Knoten 4.



Jeder Knoten hat ein nicht-negatives **Gewicht**. Wir bezeichnen das Gewicht des Knotens i ($0 \leq i < N$) mit $W[i]$.

Deine Aufgabe ist es, ein Programm zu schreiben, das Q Anfragen, die jeweils durch ein Paar von Ganzzahlen (L, R) bestimmt werden, beantwortet. Die Antwort auf die Anfrage sollte wie folgt berechnet werden:

Stellen wir uns vor, dass wir jedem Knoten des Baumes eine Ganzzahl, genannt **Koeffizient**, zuordnen. So eine Zuordnung wird durch eine Folge $C[0], \ldots, C[N-1]$ beschrieben, wobei $C[i]$ ($0 \leq i < N$) der Koeffizient ist, der Knoten i zugeordnet wird. Nennen wir diese Folge eine

Koeffizientenfolge. Beachte, dass die Glieder der Koeffizientenfolge negativ, 0 oder positiv sein können.

Für eine Anfrage (L, R) nennen wir eine Koeffizientenfolge **gültig**, wenn für jeden Knoten i ($0 \leq i < N$) die folgende Bedingung gilt: Die Summe der Koeffizienten der Knoten im Teilbaum von Knoten i ist nicht kleiner als L und nicht grösser als R.

Für eine gegebene Koeffizientenfolge $C[0], \ldots, C[N-1]$ definieren wir die **Kosten** des Knotens i als $|C[i]| \cdot W[i]$, wobei $|C[i]|$ den Betrag von $C[i]$ bezeichnet. Die **Gesamtkosten** schliesslich sind die Summe der Kosten aller Knoten. Deine Aufgabe ist es, für jede Anfrage die **minimalen Gesamtkosten** zu berechnen, die man durch die Wahl einer geeigneten gültigen Koeffizientenfolge erreichen kann.

Angaben zur Implementierung

Du sollst die folgenden zwei Funktionen implementieren:

```
void init(std::vector<int> P, std::vector<int> W)
```

- P, W: Arrays der Länge N von Ganzzahlen, die die Elternknoten und Gewichte angeben.
- Diese Funktion wird genau einmal in jedem Testfall aufgerufen, am Anfang der Interaktion zwischen dem Grader und deinem Programm.

```
long long query(int L, int R)
```

- L, R: Die Ganzzahlen, die die Anfrage beschreiben.
- Diese Funktion wird in jedem Testfall Q-mal nach dem Aufruf von `init` aufgerufen.
- Diese Funktion soll die Antwort auf die gegebene Anfrage zurückgeben.

Beschränkungen

- $1 \leq N \leq 200\,000$.
- $1 \leq Q \leq 100\,000$.
- $P[0] = -1$.
- $0 \leq P[i] < i$ für alle i mit $1 \leq i < N$.
- $0 \leq W[i] \leq 1\,000\,000$ für alle i mit $0 \leq i < N$.
- $1 \leq L \leq R \leq 1\,000\,000$ in allen Anfragen.

Subtasks

Subtask	Punkte	Zusätzliche Beschränkungen
1	10	$Q \leq 10$; $W[P[i]] \leq W[i]$ für alle i mit $1 \leq i < N$.
2	13	$Q \leq 10$; $N \leq 2\,000$.
3	18	$Q \leq 10$; $N \leq 60\,000$.
4	7	$W[i] = 1$ für alle i mit $0 \leq i < N$.
5	11	$W[i] \leq 1$ für alle i mit $0 \leq i < N$.
6	22	$L = 1$.
7	19	Keine weiteren Beschränkungen.

Beispiel

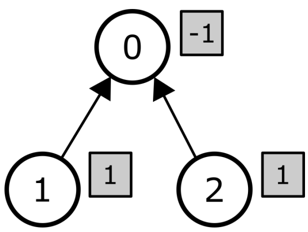
Betrachte die folgenden Aufrufe:

```
init([-1, 0, 0], [1, 1, 1])
```

Der Baum besteht aus 3 Knoten: aus der Wurzel und ihren 2 Kindern. Alle Knoten haben Gewicht 1.

```
query(1, 1)
```

In dieser Anfrage gilt $L = R = 1$, das heisst, dass die Summe der Koeffizienten in jedem Teilbaum gleich 1 sein müssen. Betrachte die Koeffizientenfolge $[-1, 1, 1]$. Der Baum und die entsprechenden Koeffizienten (in grau schattierten Rechtecken) sind hier dargestellt:



Die Summe der Koeffizienten jedes Teilbaums des Knoten i ($0 \leq i < 3$) ist 1. Dementsprechend ist die Koeffizientenfolge gültig. Die Kosten können wie folgt berechnet werden:

Für jeden Knoten i ($0 \leq i < 3$)

Knoten	Gewicht	Koeffizient	Kosten
0	1	-1	$\mid -1 \mid \cdot 1 = 1$
1	1	1	$\mid 1 \mid \cdot 1 = 1$
2	1	1	$\mid 1 \mid \cdot 1 = 1$

Die Gesamtkosten im Beispiel sind 3. Die beschriebene Koeffizientenfolge ist die einzig gültige, daher sollte der Aufruf 3 zurückgeben.

```
query(1, 2)
```

Die minimalen Kosten für diese Anfrage sind 2 und können durch die Koeffizientenfolge $[0, 1, 1]$ erreicht werden.

Beispielgrader

Eingabeformat:

```
N
P[1]  P[2] ...  P[N-1]
W[0]  W[1] ...  W[N-2] W[N-1]
Q
L[0]  R[0]
L[1]  R[1]
...
L[Q-1]  R[Q-1]
```

wobei $L[j]$ und $R[j]$ ($0 \leq j < Q$) Eingabeparameter für den j-ten Aufruf von `query` sind.

Beachte, dass die zweite Zeile der Eingabe **lediglich $\mathbf{N-1}$ Ganzzahlen** enthält, da der Beispielgrader den Wert von $P[0]$ nicht einliest.

Ausgabeformat:

```
A[0]
A[1]
...
A[Q-1]
```

wobei $A[j]$ ($0 \leq j < Q$) der Rückgabewert des j-ten Aufrufs von `query` ist.