Highest

Într-un univers alternativ, Vlad este plasat într-o versiune futuristică a Cetății Poienari, având n etaje, numerotate de la 0 la $n-1$. De la fiecare etaj i ($0 \leq i \leq n-1$), el poate urca doar în sus fie sărind pe scări, plătind 1 picătură de sânge (aceasta este valuta, pe care o folosesc pentru plăți vampirii în România), fie transformându-se într-un liliac și zburând prin canalele de ventilație, plătind pentru aceasta 2 picături de sânge. Scările îl pot duce cu până la v_i etaje în sus, în timp ce prin canalele de ventilație se poate ridica cu până la w_i etaje. Aici v și w sunt două tablouri liniare date: $v = v[0], v[1], ..., v[n-1]$ și $w = w[0], w[1], ..., w[n-1]$.

Formal, de la etajul i ($0 \leq i \leq n-1$), Vlad poate:

- sări la oricare dintre etajele de la $i+1$ până la $i+v_i$ fără a depăși etajul $n-1$, pentru un cost de 1
- zbura la oricare dintre etajele de la $i+1$ până la $i+w_i$ fără a depăși etajul $n-1$, pentru un cost de 2

Mai mult, frații lui, Radu și Mircea au propus m scenarii pentru Vlad, fiecare conținând indicii a două etaje A și B ($A \leq B$). Vlad trebuie să răspundă la m interogări: care este cea mai mică cantitate de sânge pe care el urmează să o sacrifice pentru a ajunge de la etajul A la etajul B?

Detalii de Implementare

Urmează să implementați funcția solve:

```
std::vector<int> solve(std::vector<int> &v, std::vector<int> &w,
    std::vector<std::pair<int,int>> &queries);
```

- Primește vectorii v, înălțimile săriturilor pe scări, și w, înălțimile zborurilor prin canalele de ventilație, pornind de la fiecare etaj, ambele de dimensiunea n.
- De asemenea primește interogările, un vector de perechi de întregi de dimensiunea m. Fiecare pereche conține valorile A și B după cum sunt descrise în enunț.
- Returnează un vector de dimensiunea m.

Restricții

- $1 \leq n, m \leq 500\,000$.
- $1 \leq v_i, w_i \leq n$ pentru toți $0 \leq i \leq n-1$.

- $0 \leq A \leq B \leq n-1$ în toate interogările.

Subtask-uri

1. (5 puncte) $1 \leq n \leq 300,\ 1 \leq m \leq 500\,000$
2. (7 puncte) $1 \leq n \leq 3\,000,\ 1 \leq m \leq 3\,000$
3. (11 puncte) $1 \leq n \leq 20\,000,\ 1 \leq m \leq 20\,000$
4. (44 puncte) $1 \leq n \leq 200\,000,\ 1 \leq m \leq 200\,000$
5. (8 puncte) $1 \leq n \leq 500\,000,\ 1 \leq m \leq 500\,000$ și $v_i \leq v_j$ și $w_i \leq w_j$ pentru toți $0 \leq i < j \leq n-1$
6. (25 points) Fără restricții suplimentare.

Exemple

Exemplul 1

Se consideră următorul apel:

```
solve({2, 3, 1, 1, 1, 1, 2}, {3, 4, 1, 2, 1, 2, 2},
    {{0, 4}, {0, 5}, {0, 6}})
```

Aici avem $n = 7$ și 3 interogări, $v = [2, 3, 1, 1, 1, 1, 2]$ și $w = [3, 4, 1, 2, 1, 2, 2]$

Pentru prima interogare $(0, 4)$, Vlad trebuie să facă două sărituri de cost 1: 0 la 1 (chiar dacă el poate sări la etajul 2, etajul 1 îl poate duce mai departe), apoi de la 1 la 4. Costul total: $1 + 1 = 2$.

Pentru cea de a doua interogare $(0, 5)$, există 2 drumuri optimale: De la 0 la 1 (cost 1), de la 1 la 4 (cost 1), de la 4 la 5 (cost 1); al doilea traseu este de la 0 la 1 (cost 1), de la 1 la 5 (cost 2). Cost total: $1 + 1 + 1 = 1 + 2 = 3$.

Pentru a treia interogare $(0, 6)$, un exemplu de drum cu cost 4 ar fi de la 0 la 1 (cost 1), de la 1 la 5 (cost 2), de la 5 la 6 (cost 1). Cost total: $1 + 2 + 1 = 4$

Astfel, vectorul returnat de funcție va fi:

```
{2, 3, 4}
```

Exemplul 2

Se consideră următorul apel:

```
solve({1, 1, 1, 2, 3, 2, 1, 1, 2, 3}, {2, 4, 1, 4, 1, 4, 1, 3, 2, 3},
    {{3, 9}, {0, 9}, {0, 7}, {0, 4}, {3, 5}})
```

Acestea sunt drumurile optimale pentru interogări:

$(3,9)$: de la 3 la 5 (cost 1), de la 5 la 9 (cost 2) $\Longrightarrow$ total: 3

$(0,9)$: de la 0 la 1 (cost 1), de la 1 la 5 (cost 2), de la 5 la 9 (cost 2) $\Longrightarrow$ total: 5

$(0,7)$: de la 0 la 1 (cost 1), de la 1 la 5 (cost 2), de la 5 la 7 (cost 1) $\Longrightarrow$ total: 4

$(0,4)$: de la 0 la 1 (cost 1), de la 1 la 4 (cost 2) $\Longrightarrow$ total: 3

$(3,5)$: de la 3 la 5 (cost 1) $\Longrightarrow$ total: 1

Astfel, vectorul returnat de funcție va fi:

```
{3, 5, 4, 3, 1}
```

Grader Local

Graderul local citește datele din input în următorul format:

- linie 1: n
- linie 2: $v[0]$ $v[1]$ $\ldots$ $v[n-1]$
- linie 3: $w[0]$ $v[1]$ $\ldots$ $w[n-1]$
- linie 4: m
- linie $5+i (0 \leq i < n)$: A B

și afișează m linii, rezultatul apelului funcției `solve`.