Nile

Celin želi prevesti N gajdica preko Nila u svoju palaču. Gajdice su indeksirane od 0 do $N-1$. Starost gajdice i ($0 \leq i < N$), izražena u danima iznosi $W[i]$.

Za prijevoz je nabavio posebne venecijanske lađe. Svaka lađa može prenjeti **maksimalno dvije** gajdice.

- U slučaju da je sama, bilo koja gajdica može biti na brodu, tj. nema ograničenja.
- U slučaju da će na brodu biti dvije stvar je malo delikatnija. Celinu dob nije bitna no znano je da ako je sklop u glavi različit doći će do svađe. Formalno, dvije gajdice p i q ($0 \leq p < q < N$) mogu biti u istom brodu ako i samo ako su slične starosti, tj. ako je $|W[p] - W[q]| \leq D$.

Za prijevoz gajdica treba osigurati određenu količinu hrane koja ovisi o tome koliko je gajdica na brodu. Količina hrane za gajicu i ($0 \leq i < N$) je:

- $A[i]$, ako je sama na brodu, or
- $B[i]$, ako je na brodu s nekom drugom.

Naravno ako su dvije gajdice na brodu potrebno je osigurati hranu za obje. Formalnije, npr. ako pošalje dvije rudlave gajdice p i q ($0 \leq p < q < N$) u isti brod, mora nabaviti $B[p] + B[q]$ hrane.

Slanje gajdice same je uvijek trošnije jer će od nervoze i samoće više ogladnjeti, dakle uvijek je $B[i] < A[i]$ za sve i t.d. $0 \leq i < N$.

Nažalost rijeka je nepredvidiva pa će Celin isplanirati Q scenarija. Zadatak je odgovoriti Q pitanja naznačena od 0 do $Q-1$. Pitanja su opisana nizom E duljine Q. Odgovor na pitanje j ($0 \leq j < Q$) je minimalni trošak hrane za prijevoz svih N gajdica (čak i onih ne rudlavih), kada je vrijednost D-a jednaka $E[j]$.

Implementation Details

You should implement the following procedure.

```
std::vector<long long> calculate_costs(
    std::vector<int> W, std::vector<int> A,
    std::vector<int> B, std::vector<int> E)
```

- W, A, B: arrays of integers of length N, describing the weights of the artifacts and the costs of transporting them.
- E: an array of integers of length Q describing the value of D for each question.
- This procedure should return an array R of Q integers containing the minimum total cost of transporting the artifacts, where $R[j]$ gives the cost when the value of D is $E[j]$ (for each j such that $0 \leq j < Q$).
- This procedure is called exactly once for each test case.

Constraints

- $1 \leq N \leq 100\,000$
- $1 \leq Q \leq 100\,000$
- $1 \leq W[i] \leq 10^9$ for each i such that $0 \leq i < N$
- $1 \leq B[i] < A[i] \leq 10^9$ for each i such that $0 \leq i < N$
- $1 \leq E[j] \leq 10^9$ for each j such that $0 \leq j < Q$

Subtasks

Subtask	Score	Additional Constraints
1	6	$Q \leq 5$; $N \leq 2000$; $W[i] = 1$ for each i such that $0 \leq i < N$
2	13	$Q \leq 5$; $W[i] = i + 1$ for each i such that $0 \leq i < N$
3	17	$Q \leq 5$; $A[i] = 2$ and $B[i] = 1$ for each i such that $0 \leq i < N$
4	11	$Q \leq 5$; $N \leq 2000$
5	20	$Q \leq 5$
6	15	$A[i] = 2$ and $B[i] = 1$ for each i such that $0 \leq i < N$
7	18	No additional constraints.

Example

Consider the following call.

```
calculate_costs([15, 12, 2, 10, 21],
                [5, 4, 5, 6, 3],
                [1, 2, 2, 3, 2],
                [5, 9, 1])
```

In this example we have $N = 5$ artifacts and $Q = 3$ questions.

In the first question, $D = 5$. You can send artifacts 0 and 3 in one boat (since $|15 - 10| \leq 5$) and the remaining artifacts in separate boats. This yields the minimum cost of transporting all the

artifacts, which is $1 + 4 + 5 + 3 + 3 = 16$.

In the second question, $D = 9$. You can send artifacts 0 and 1 in one boat (since $|15 - 12| \leq 9$) and send artifacts 2 and 3 in one boat (since $|2 - 10| \leq 9$). The remaining artifact can be sent in a separate boat. This yields the minimum cost of transporting all the artifacts, which is $1 + 2 + 2 + 3 + 3 = 11$.

In the final question, $D = 1$. You need to send each artifact in its own boat. This yields the minimum cost of transporting all the artifacts, which is $5 + 4 + 5 + 6 + 3 = 23$.

Hence, this procedure should return $[16, 11, 23]$.

Sample Grader

Input format:

```
N
W[0] A[0] B[0]
W[1] A[1] B[1]
...
W[N-1] A[N-1] B[N-1]
Q
E[0]
E[1]
...
E[Q-1]
```

Output format:

```
R[0]
R[1]
...
R[S-1]
```

Here, S is the length of the array R returned by `calculate_costs`.