#### **Time limit:** 2.0s **Memory limit:** 1G

You want to transport N artifacts through the Nile. The artifacts are numbered from 0 to N-1. The weight of artifact i ( $0 \le i < N$ ) is W[i].

To transport the artifacts, you use specialized boats. Each boat can carry at most two artifacts.

- If you decide to put a single artifact in a boat, the artifact weight can be arbitrary.
- If you want to put two artifacts in the same boat, you have to make sure the boat is balanced evenly. Specifically, you can send artifacts p and q ( $0 \le p < q < N$ ) in the same boat only if the absolute difference between their weights is at most D, that is  $|W[p] W[q]| \le D$ .

To transport an artifact, you have to pay a cost that depends on the number of artifacts carried in the same boat. The cost of transporting artifact i ( $0 \le i < N$ ) is:

- A[i], if you put the artifact in its own boat, or
- B[i], if you put it in a boat together with some other artifact.

Note that in the latter case, you have to pay for both artifacts in the boat. Specifically, if you decide to send artifacts p and q ( $0 \le p < q < N$ ) in the same boat, you need to pay B[p] + B[q].

Sending an artifact in a boat by itself is always more expensive than sending it with some other artifact sharing the boat with it, so B[i] < A[i] for all i such that  $0 \le i < N$ .

Unfortunately, the river is very unpredictable and the value of D changes often. Your task is to answer Q questions numbered from 0 to Q-1. The questions are described by an array E of length Q. The answer to question j (  $0 \le j < Q$ ) is the minimum total cost of transporting all N artifacts, when the value of D is equal to 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  $0 \le j < Q$ ).
- This procedure is called exactly once for each test case.

#### **Constraints**

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

### **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.

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  $|15-10| \le 5$ ) 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 <code>calculate\_costs</code> .

# **Attachment Package**

The sample grader along with sample test cases are available here.