Overtaking 超車

從布達佩斯機場到福拉斯酒店有一條單線單行車道。這條路長 L 公里。

在 IOI 2023 活動期間，有 $N+1$ 輛接駁巴士在這條路上行駛。巴士從 0 到 N 編號。 巴士 i ($0 \le i < N$) 被安排在活動的第 $T[i]$ 秒離開機場，並且每 $W[i]$ 秒行駛 1 公里。 巴士 N 是一輛備用巴士，可以每 X 秒行駛 1 公里。 但它離開機場的時間 Y 還沒有確定。

一般情況下，在路上是不允許超車的，但是巴士可以在**排序站點**超車。

在路上有 M ($M > 1$) 個排序站點，從 0 到 $M-1$ 編號，位於路上不同的位置。

排序站點 j ($0 \le j < M$) 距離機場 $S[j]$ 公里。

排序站點按照距離機場遞增的順序排序，即對於每個 $0 \le j \le M-2$， $S[j] < S[j+1]$。

第一個排序站點是在機場，最後一個是在酒店，即 $S[0] = 0$ 且 $S[M-1] = L$。

每輛巴士以最大速度行駛，除非它追上了前面行駛的速度較慢的巴士，此時它們會被迫以較慢巴士的速度行駛，直到它們到達下一個排序站點。 在那裡，速度較快的巴士將超過速度較慢的巴士。

具體而言，對於每個 i 和 j，其中 $0 \le i \le N$ 且 $0 \le j < M$，定義巴士 i **到達**排序站點 j 的時間 $t_{i,j}$（以秒為單位）如下: 對於每個 $0 \le i < N$，令 $t_{i,0} = T[i]$，並且令 $t_{N,0} = Y$。 對於每個 $0 < j < M$ ：

- 定義巴士 i 在排序站點 j 的**預計到達時間**（以秒為單位），表示巴士 i 如果從它到達排序站點 $j-1$ 的時間開始以全速行駛，將在什麼時間到達排序站點 j。即
  - 對於每個 $0 \le i < N$， $e_{i,j} = t_{i,j-1} + W[i] \cdot (S[j] - S[j-1])$，
  - $e_{N,j} = t_{N,j-1} + X \cdot (S[j] - S[j-1])$。
- 巴士 i 在排序站點 j 到達的時間是巴士 i 和每輛在排序站點 $j-1$ 比巴士 i 先到達的其他巴士的預計到達時間的**最大值**。具體而言，令 $t_{i,j}$ 是 $e_{i,j}$ 和每個 $e_{k,j}$ 的最大值，其中 $0 \le k \le N$ 且 $t_{k,j-1} < t_{i,j-1}$。

IOI 的組織者希望安排備用巴士（巴士 N）的行程。 你的任務是回答組織者的 Q 個問題，問題的形式如下：給定備用巴士（巴士 N）預計從機場出發的時間 Y （以秒為單位），它將在什麼時間到達酒店?

實現細節

你需要實現以下兩個函數。

```
void init(int L, int N, int64[] T, int[] W, int X, int M, int[] S)
```

- L： 路的長度。
- N： 非備用巴士的數量。
- T： 長度為 N 的數組，描述非備用巴士從機場出發的時間。
- W： 長度為 N 的數組，描述非備用巴士的最大速度。
- X： 備用巴士行駛 1 公里所需的時間。
- M： 排序站點的數量。
- S： 長度為 M 的數組，描述排序站點距離機場的距離。
- 此函數在調用 `arrival_time` 之前，對每個測試用例只調用一次。

```
int64 arrival_time(int64 Y)
```

- Y： 備用巴士（巴士 N）預計從機場出發的時間。
- 此函數應該返回借用巴士 N 到達酒店的時間。
- 此函數將被調正好 Q 次。

範例

考慮以下的調用順序：

```
init(6, 4, [20, 10, 40, 0], [5, 20, 20, 30], 10, 4, [0, 1, 3, 6])
```

忽略尚未安排的巴士 4，下表顯示了每個排序站點的非保留巴士的預期和實際到達時間：

i		$t_{i,0}$		$e_{i,1}$	$t_{i,1}$		$e_{i,2}$	$t_{i,2}$		$e_{i,3}$	$t_{i,3}$
0		20		25	30		40	40		55	55
1		10		30	30		70	70		130	130
2		40		60	60		100	100		160	180
3		0		30	30		90	90		180	180

排序站點 0 的到達時間是巴士計劃離開機場的時間。也就是說，對於 $0 \leq i \leq 3$， $t_{i,0} = T[i]$。

計算排序站點 1 的預期和實際到達時間如下：

- 排序站點 1 的預期到達時間：
  - 巴士 0 ： $e_{0,1} = t_{0,0} + W[0] \cdot (S[1] - S[0]) = 20 + 5 \cdot 1 = 25$。
  - 巴士 1 ： $e_{1,1} = t_{1,0} + W[1] \cdot (S[1] - S[0]) = 10 + 20 \cdot 1 = 30$。
  - 巴士 2 ： $e_{2,1} = t_{2,0} + W[2] \cdot (S[1] - S[0]) = 40 + 20 \cdot 1 = 60$。
  - 巴士 3 ： $e_{3,1} = t_{3,0} + W[3] \cdot (S[1] - S[0]) = 0 + 30 \cdot 1 = 30$。
- 排序站點 1 的到達時間：
  - 巴士 1 和巴士 3 比巴士 0 更早到達排序站點 0，所以 $t_{0,1} = \max([e_{0,1}, e_{1,1}, e_{3,1}]) = 30$。
  - 巴士 3 比巴士 1 更早到達排序站點 0，所以 $t_{1,1} = \max([e_{1,1}, e_{3,1}]) = 30$。

- 巴士 0、巴士 1 和巴士 3 比巴士 2 更早到達排序站點 0，所以 $t_{2,1}=\max([e_{0,1},e_{1,1},e_{2,1},e_{3,1}])=60$。
- 沒有巴士比巴士 3 更早到達排序站點 0，所以 $t_{3,1}=\max([e_{3,1}])=30$。

```
arrival_time(0)
```

巴士 4 需要 10 秒鐘才能行駛 1 公里，現在預計在第 0 秒離開機場。 在這種情況下，下表顯示了每個巴士的到達時間。關於非保留巴士的預期和實際到達時間，唯一的變化是用下劃線標出的。

i		$t_{i,0}$		$e_{i,1}$	$t_{i,1}$		$e_{i,2}$	$t_{i,2}$		$e_{i,3}$	$t_{i,3}$
0		20		25	30		40	40		55	$\underline{60}$
1		10		30	30		70	70		130	130
2		40		60	60		100	100		160	180
3		0		30	30		90	90		180	180
4		0		10	10		30	30		60	60

我們可以看到巴士 4 在第 60 秒到達酒店。 因此，該程序應該返回 60。

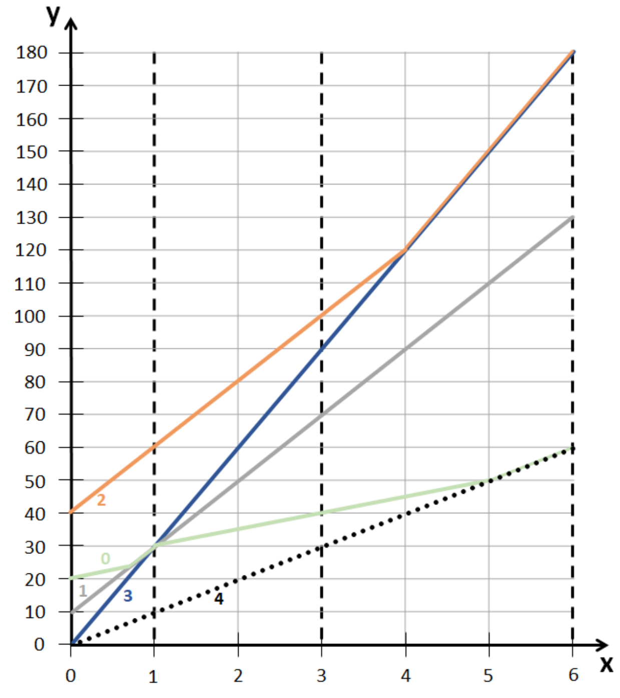
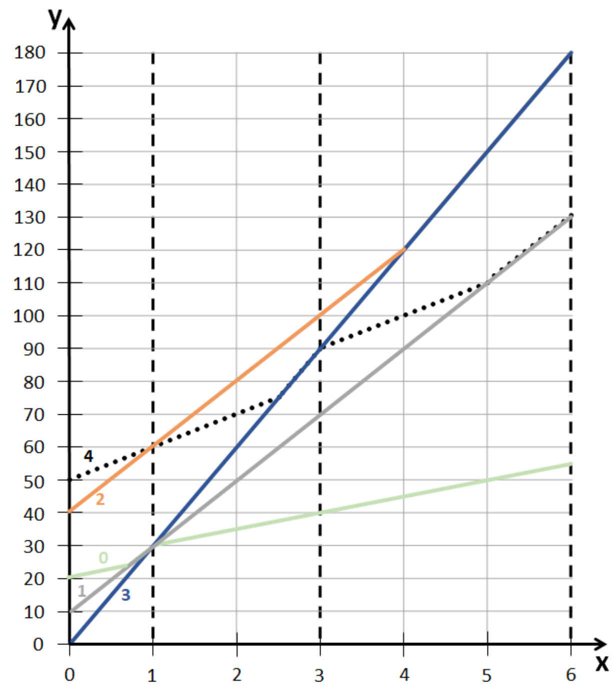
```
arrival_time(50)
```

巴士 4 現在預計在第 50 秒離開機場。 在這種情況下，與初始表格相比，非保留巴士的到達時間沒有變化。 下表顯示了到達時間。

i		$t_{i,0}$		$e_{i,1}$	$t_{i,1}$		$e_{i,2}$	$t_{i,2}$		$e_{i,3}$	$t_{i,3}$
0		20		25	30		40	40		55	55
1		10		30	30		70	70		130	130
2		40		60	60		100	100		160	180
3		0		30	30		90	90		180	180
4		50		60	60		80	90		120	130

巴士 4 在排序站點 1 超過較慢的巴士 2，因為它們同時到達。 接下來，巴士 4 在排序站點 1 和排序站點 2 之間與巴士 3 一起行駛，使得巴士 4 在第 90 秒到達排序站點 2，而不是第 80 秒。 離開排序站點 2 後，巴士 4 與巴士 1 一起行駛，直到到達酒店。 巴士 4 在第 130 秒到達酒店。 因此，該程序應該返回 130。

我們可以繪製每輛巴士到達機場不同距離所需的時間。 圖表的 x 軸表示距離機場的距離（以公里為單位），y 軸表示時間（以秒為單位）。 垂直虛線標記了排序站點的位置。 不同的實線（附帶巴士索引）表示四輛非備用巴士。 黑色虛線表示備用巴士。

<table>
<tr><th><code>arrival_time(0)</code></th><th><code>arrival_time(50)</code></th></tr>
<tr><td>
</td><td>
</td></tr>
</table>

約束條件

- $1 \le L \le 10^9$
- $1 \le N \le 1\,000$
- $0 \le T[i] \le 10^{18}$（對於每個 i，滿足 $0 \le i < N$）
- $1 \le W[i] \le 10^9$（對於每個 i，滿足 $0 \le i < N$）
- $1 \le X \le 10^9$
- $2 \le M \le 1\,000$
- $0 = S[0] < S[1] < \cdots < S[M-1] = L$
- $1 \le Q \le 10^6$
- $0 \le Y \le 10^{18}$

子任務

1. （9 分）$N = 1, Q \le 1\,000$
2. （10 分）$M = 2, Q \le 1\,000$
3. （20 分）$N, M, Q \le 100$
4. （26 分）$Q \le 5\,000$
5. （35 分）無額外限制

範例評分程式

範例評分程式按照以下格式讀取輸入：

- 第 1 行： $L\ N\ X\ M\ Q$
- 第 2 行： $T[0]\ T[1]\ \ldots\ T[N-1]$
- 第 3 行： $W[0]\ W[1]\ \ldots\ W[N-1]$
- 第 4 行： $S[0]\ S[1]\ \ldots\ S[M-1]$
- 第 $5+k$ 行（$0 \leq k < Q$）：問題 k 的 Y

範例評分程式以以下格式打印你的答案：

- 第 $1+k$ 行（$0 \leq k < Q$）：問題 k 的 `arrival_time` 的返回值