Sorpassi

La strada tra l'aeroporto di Budapest e l'Hotel Forrás è composta da un'unica corsia a senso unico lunga L km.

Durante le IOI 2023 bis (che devono ancora svolgersi), $N+1$ bus percorreranno la strada, numerati da 0 a N. Per $0 \leq i < N$, il bus i partirà dall'aeroporto al $T[i]$-esimo secondo dell'evento, e percorre 1 km in $W[i]$ secondi. Il bus N è di riserva: percorre 1 km in X secondi, e il momento Y in cui partirà non è ancora stato deciso.

Anche se i bus non si possono sorpassare lungo la strada, possono farlo in $M > 1$ punti particolari detti **stazioni di smistamento**, numerati da 0 a $M-1$. La stazione j ($0 \leq j < M$) è situata sulla strada a $S[j]$ km dalla partenza. La prima stazione è alla partenza e l'ultima all'arrivo, per cui $S[0] = 0$ e $S[M-1] = L$, e anche eventuali altre stazioni intermedie sono elencate in ordine crescente.

Ogni bus viaggia alla sua massima velocità, a meno che non raggiunga un bus più lento che viaggia davanti a lui, nel qual caso viaggia esattamente insieme al bus più lento fino alla successiva stazione di smistamento. In tale stazione, gli autobus che arrivano insieme ne escono in ordine dal più veloce (per primo) al più lento (per ultimo).

Formalmente, per ogni i e j tale che $0 \leq i \leq N$ e $0 \leq j < M$, il tempo $t_{i,j}$ (in secondi) in cui il bus i **arriva** alla stazione j è definito come segue. Sia $t_{i,0} = T[i]$ per ogni $0 \leq i < N$, e sia $t_{N,0} = Y$. Per ogni j tale che $0 < j < M$:

- Sia $e_{i,j}$ il **tempo atteso di arrivo** (in secondi) del bus i alla stazione j, pari al tempo in cui il bus i arriverebbe alla stazione j se viaggiasse a massima velocità dal momento in cui è arrivato alla stazione $j-1$. Quindi, sia
  - $e_{i,j} = t_{i,j-1} + W[i] \cdot (S[j] - S[j-1])$ per ogni $0 \leq i < N$, e
  - $e_{N,j} = t_{N,j-1} + X \cdot (S[j] - S[j-1])$.
- Il bus i arriva alla stazione j al *massimo* dei tempi attesi di arrivo del bus i e di ogni altro bus che passa dalla stazione $j-1$ strettamente prima del bus i. Formalmente, $t_{i,j}$ è il massimo tra $e_{i,j}$ e ogni $e_{k,j}$ per cui $0 \leq k \leq N$ e $t_{k,j-1} < t_{i,j-1}$.

Gli organizzatori devono decidere quando far partire il bus N. Devi quindi rispondere a Q domande degli organizzatori della forma: dato il tempo Y (in secondi) in cui il bus N parte dall'aeroporto, in quale tempo arriverà all'hotel?

Note di implementazione

Devi implementare la seguente funzione:

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

- L: la lunghezza della strada.
- N: il numero di bus non di riserva.
- T: un array di lunghezza N che contiene i momenti in cui i bus non di riserva partono dall'aeroporto.
- W: un array di lunghezza N che contiene le velocità dei bus non di riserva.
- X: il tempo che il bus N (di riserva) impiega a percorrere 1 km.
- M: il numero di stazioni di smistamento.
- S: un array di lunghezza M che contiene le distanze delle stazioni di smistamento dall'aeroporto.
- Questa funzione verrà chiamata esattamente una volta per testcase, prima di qualsiasi chiamata ad `arrival_time`.

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

- Y: L'istante in cui il bus N potrebbe partire dall'aeroporto.
- Questa funzione deve restituire l'istante in cui il bus N arriverebbe all'hotel.
- Questa funzione verrà chiamata esattamente Q volte.

Esempio

Considera la seguenza di chiamate:

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

Ignorando il bus 4 (che non è stato ancora pianificato), questi sono i tempi previsti ed effettivi di arrivo dei bus in ogni stazione di smistamento:

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

I tempi di arrivo alla stazione 0 sono gli istanti in cui i bus partono dall'aeroporto. Quindi, $t_{i,0} = T[i]$ per $0 \leq i \leq 3$.

I tempi previsti ed effettivi di arrivo alla stazione 1 sono calcolati come segue:

- Tempi previsti di arrivo:
  - Bus 0: $e_{0,1} = t_{0,0} + W[0] \cdot (S[1] - S[0]) = 20 + 5 \cdot 1 = 25$.
  - Bus 1: $e_{1,1} = t_{1,0} + W[1] \cdot (S[1] - S[0]) = 10 + 20 \cdot 1 = 30$.
  - Bus 2: $e_{2,1} = t_{2,0} + W[2] \cdot (S[1] - S[0]) = 40 + 20 \cdot 1 = 60$.
  - Bus 3: $e_{3,1} = t_{3,0} + W[3] \cdot (S[1] - S[0]) = 0 + 30 \cdot 1 = 30$.
- Tempi effettivi:
  - I bus 1 e 3 partono dalla stazione 0 prima del bus 0, quindi $t_{0,1} = \max([e_{0,1}, e_{1,1}, e_{3,1}]) = 30$.
  - Il bus 3 parte dalla stazione 0 prima del bus 1, quindi $t_{1,1} = \max([e_{1,1}, e_{3,1}]) = 30$.
  - I bus 0, 1 e 3 partono dalla stazione 0 prima del bus 2, quindi $t_{2,1} = \max([e_{0,1}, e_{1,1}, e_{2,1}, e_{3,1}]) = 60$.
  - Nessun bus parte dalla stazione 0 prima del bus 3, quindi $t_{3,1} = \max([e_{3,1}]) = 30$.

```
arrival_time(0)
```

Il bus 4 percorre 1 km in 10 secondi e parte dall'aeroporto al secondo 30. In questo caso, i tempi di arrivo sono i seguenti. L'unica modifica ai tempi degli altri bus è sottolineata.

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

Il bus 4 arriva all'hotel al secondo 60, quindi la funzione deve restituire 60.

```
arrival_time(50)
```

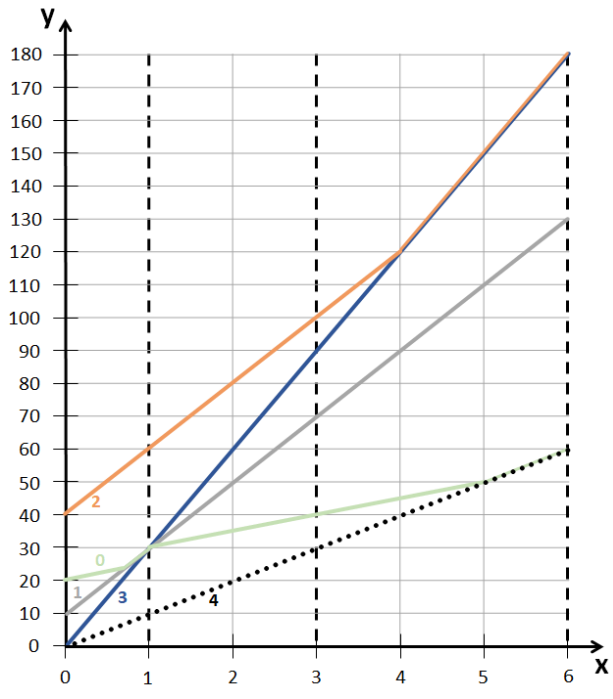
Ora il bus 4 parte al secondo 50. In questo caso, non ci sono modifiche ai tempi di arrivo degli altri bus. I tempi sono mostrati di seguito.

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

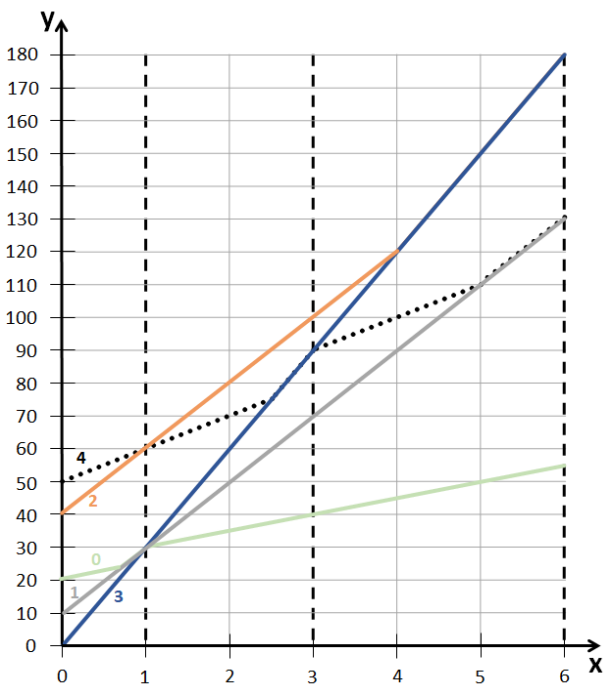
Il bus 4 supera il bus 2, più lento, alla stazione 1, dato che arrivano nello stesso momento. Poi, il bus 4 viaggia insieme al bus 3 tra le stazioni 1 e 2, arrivando alla stazione 2 al secondo 90 invece che 80. Lasciata la stazione 2, il bus 4 viaggia insieme al bus 1 fino all'albergo, dove arriva al secondo 130. La funzione deve quindi restituire 130.

Possiamo tracciare il tempo impiegato dai bus per percorrere la strada. L'asse x dei grafici rappresenta la distanza dall'aeroporto (in km) e l'asse y rappresenta il tempo trascorso (in secondi). Le linee verticali tratteggiate indicano le posizioni delle stazioni di interscambio. Le linee continue indicano i percorsi degli autobus già pianificati, mentre la linea puntinata rappresenta l'autobus N.

`arrival_time(0)`



`arrival_time(50)`



Assunzioni

- $1 \leq L \leq 10^9$.
- $1 \leq N \leq 1\,000$.
- $0 \leq T[i] \leq 10^{18}$ per ogni $0 \leq i < N$.
- $1 \leq W[i] \leq 10^9$ per ogni $0 \leq i < N$.
- $1 \leq X \leq 10^9$.
- $2 \leq M \leq 1\,000$.
- $0 = S[0] < S[1] < \cdots < S[M-1] = L$.
- $1 \leq Q \leq 10^6$.
- $0 \leq Y \leq 10^{18}$.

Subtask

1. (9 punti) $N = 1$, $Q \leq 1\,000$.
2. (10 punti) $M = 2$, $Q \leq 1\,000$
3. (20 punti) $N, M, Q \leq 100$.
4. (26 punti) $Q \leq 5\,000$.
5. (35 punti) Nessuna limitazione aggiuntiva.

Grader di esempio

Il grader di esempio legge l'input nel seguente formato:

- riga 1: L N X M Q
- riga 2: $T[0]$ $T[1]$ $\ldots$ $T[N-1]$
- riga 3: $W[0]$ $W[1]$ $\ldots$ $W[N-1]$
- riga 4: $S[0]$ $S[1]$ $\ldots$ $S[M-1]$
- riga $5+k$ ($0 \leq k < Q$): Y per la query k

Il grader di esempio stampa le tue risposte nel seguente formato:

- riga $1+k$ ($0 \leq k < Q$): il valore restituito da `arrival_time` per la query k