Overtaking

Între aeroportul Budapesta și Hotelul Forrás există o șosea cu o singură bandă și sens unic, de L kilometri.

În timpul IOI 2023, $N+1$ autobuze circulă pe această șosea. Autobuzele sunt numerotate cu numere de la 0 la N. Autobuzul i ($0 \leq i < N$) este programat să părăsească aeroportul la secunda $T[i]$ a evenimentului și poate parcurge 1 kilometru în $W[i]$ secunde. Autobuzul N este un autobuz de rezervă care poate parcurge 1 kilometru în X secunde. Momentul de timp Y când acesta va pleca de la aeroport nu a fost decis încă.

Depășirile sunt interzise pe această șosea în general, însă autobuzele se pot depăși în așa-numitele **stații de sortare**. Există M ($M > 1$) stații de sortare pe șosea, numerotate cu numere de la 0 la $M-1$. Stația de sortare j ($0 \leq j < M$) este amplasată de-a lungul șoselei la $S[j]$ kilometri de aeroport. Stațiile de sortare sunt sortate în ordine crescătoare a distanței față de aeroport, adică, $S[j] < S[j+1]$ pentru fiecare $0 \leq j \leq M-2$. Prima stație de sortare este chiar în aeroport, iar ultima chiar la hotel, mai exact, $S[0] = 0$ și $S[M-1] = L$.

Autobuzele circulă la viteza lor maximă până când ajunge în dreptul unui autobuz mai încet care circulă în fața lui pe șosea, caz în care se formează o coloană care e forțată să circule la aceeași viteză a autobuzului mai încet până la următoarea stație de sortare. Odată ajunse acolo, autobuzele mai rapide vor depăși autobuzele mai încete.

Formal, pentru fiecare i și j cu $0 \leq i \leq N$ și $0 \leq j < M$, timpul $t_{i,j}$ (în secunde) când autobuzul i **ajunge la** stația de sortare j se definește astfel. Fie $t_{i,0} = T[i]$ pentru orice $0 \leq i < N$ și fie $t_{N,0} = Y$. Pentru fiecare j cu $0 < j < M$:

- Definim **timpul așteptat de sosire** (în secunde) al autobuzului i la stația de sortare j, notat $e_{i,j}$, ca fiind momentul la care autobuzul i ar ajunge la stația de sortare j dacă ar circula la viteza sa maximă din momentul în care ajunge la stația de sortare $j-1$. Mai exact, fie
  - $e_{i,j} = t_{i,j-1} + W[i] \cdot (S[j] - S[j-1])$ pentru orice $0 \leq i < N$ și
  - $e_{N,j} = t_{N,j-1} + X \cdot (S[j] - S[j-1])$.
- Autobuzul i ajunge la stația de sortare j la *maximul* dintre timpii așteptați de sosire ai autobuzului i și a oricărui alt autobuz care ajunsese la stația de sortare $j-1$ mai devreme decât autobuzul i. Formal, fie $t_{i,j}$ maximul dintre $e_{i,j}$ și fiecare $e_{k,j}$ pentru care $0 \leq k \leq N$ și $t_{k,j-1} < t_{i,j-1}$.

Organizatorii IOI vor să programeze autobuzul de rezervă (autobuzul N). Misiunea ta este să răspunzi la Q întrebări din partea organizatorilor, de forma: dându-se momentul de timp Y (în secunde) când autobuzul de rezervă se presupune să părăsească aeroportul, la ce moment va ajunge la hotel?

Detalii de implementare

Sarcina ta este să implementezi următoarele proceduri.

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

- L: lungimea drumului.
- N: numărul de autobuze planificate pentru transfer.
- T: un vector de lungime N care descrie timpii la care autobuzele $0, \ldots, N-1$ sunt planificate să plece de la aeroport.
- W: un vector de lungime N ce descrie vitezele maxime ale autobuzelor $0, \ldots, N-1$.
- X: timpul necesar pentru autobuzul de rezervă pentru a parcurge 1 kilometru.
- M: numărul stațiilor de sortare.
- S: un vector de lungime M ce descrie distanțele stațiilor de sortare de la aeroport.
- Această procedură este apelată o singură dată pentru fiecare test, înainte de orice apel al funcției `arrival_time`.

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

- Y: timpul la care autobuzul de rezervă (autobuzul N) este de presupus că va pleca de la aeroport.
- Această procedură trebuie să returneze timpul la care autobuzul de rezervă ar trebui să ajungă la hotel.
- Această procedură este apelată de exact Q ori.

Exemplu

Considerăm următoarea secvență de apeluri:

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

Ignorând autobuzul 4 (care nu a fost încă programat), următorul tabel arată timpii așteptați și actuali de sosire pentru autobuzele $0, 1, 2$ și 3 la fiecare stație de sortare:

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

Timpii de sosire la stația 0 sunt timpii în care autobuzele sunt planificate să plece de la aeroport. Cu alte cuvinte, $t_{i,0} = T[i]$ pentru $0 \leq i \leq 3$.

Timpii așteptați și actuali de sosire la stația de sortare 1 se calculează astfel:

- Timpii așteptați de sosire la stația 1:
  - Autobuz 0: $e_{0,1} = t_{0,0} + W[0] \cdot (S[1] - S[0]) = 20 + 5 \cdot 1 = 25$.
  - Autobuz 1: $e_{1,1} = t_{1,0} + W[1] \cdot (S[1] - S[0]) = 10 + 20 \cdot 1 = 30$.
  - Autobuz 2: $e_{2,1} = t_{2,0} + W[2] \cdot (S[1] - S[0]) = 40 + 20 \cdot 1 = 60$.
  - Autobuz 3: $e_{3,1} = t_{3,0} + W[3] \cdot (S[1] - S[0]) = 0 + 30 \cdot 1 = 30$.
- Timpii actuali de sosire la stația 1:
  - Autobuzele 1 and 3 ajung în stația 0 mai devreme decât 0, deci $t_{0,1} = \max(e_{0,1}, e_{1,1}, e_{3,1}) = 30$.
  - Autobuzul 3 ajunge la stația 0 mai devreme decât autobuzul 1, deci $t_{1,1} = \max(e_{1,1}, e_{3,1}) = 30$.
  - Autobuzele 0, 1 și 3 ajung la stația 0 mai devreme decât autobuzul 2, deci $t_{2,1} = \max(e_{0,1}, e_{1,1}, e_{2,1}, e_{3,1}) = 60$.
  - Niciun autobuz nu ajunge la stația 0 mai devreme ca autobuzul 3, deci $t_{3,1} = \max(e_{3,1}) = 30$.

```
arrival_time(0)
```

Autobuzului 4 îi ia 10 secunde să parcurgă 1 kilometru și este programat să părăsească aeroportul la secunda 0. În acest caz, următorul tabel arată timpii de sosire pentru fiecare autobuz. Singura schimbare în legătură cu timpii așteptați și actuali ai autobuzelor non-rezervă este subliniată.

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

Putem vedea că autobuzul 4 ajunge la hotel la secunda 60. Prin urmare, procedura trebuie să returneze 60.

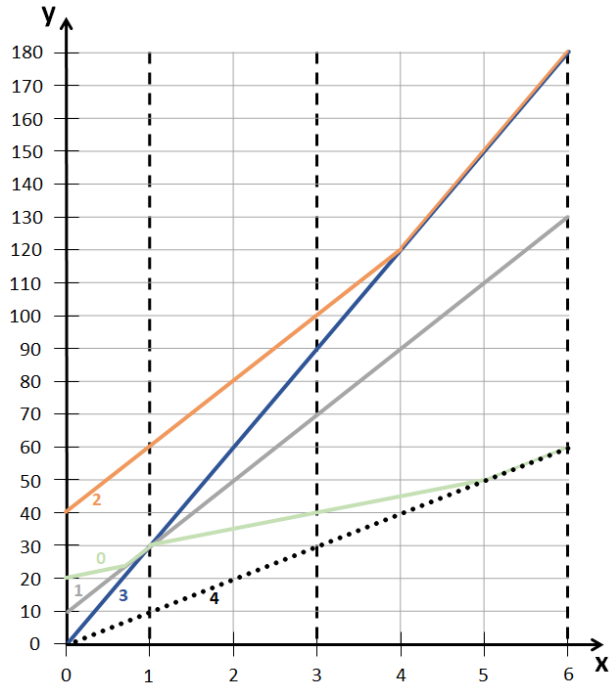
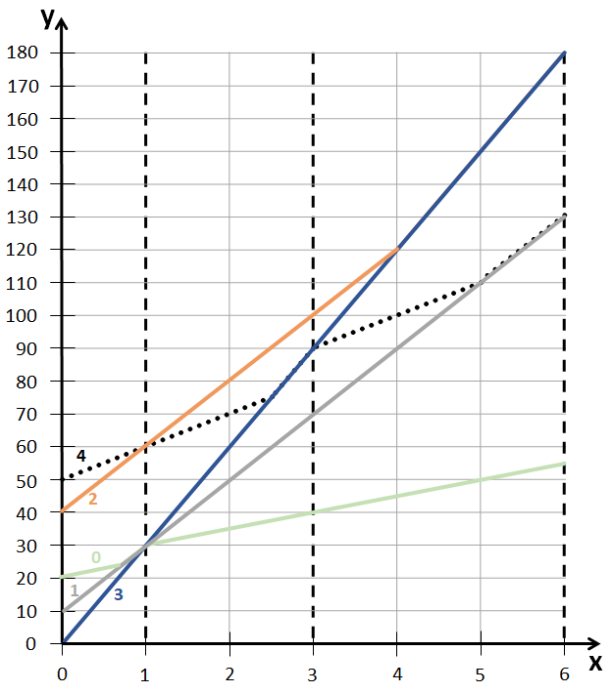
```
arrival_time(50)
```

Autobuzul 4 este programat să părăsească aeroportul la secunda 50. În acest caz, timpii de sosire a celorlalte autobuze nu se schimbă față de tabelul inițial. Timpii de sosire se pot vedea în următorul tabel.

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

Autobuzul 4 depășește autobuzul 2 la stația de sortare 1, unde ajung în același timp. Mai apoi, autobuzul 4 formează o coloană cu autobuzul 3 între stațiile 1 și 2, făcând ca autobuzul 4 să ajungă la stația 2 la secunda 90 în loc de 80. După părăsirea stației 2, autobuzul 4 formează o coloană cu autobuzul 1 până ce ajung la hotel. Autobuzul 4 ajunge la hotel la secunda 130. Prin urmare, procedura va trebui să returneze 130.

Putem reprezenta grafic timpul pe care îl petrece fiecare autobuz pentru a parcurge fiecare kilometru dinspre aeroport. Axa x a acestui grafic reprezintă distanța față de aeroport (în kilometri), iar axa y reprezintă timpul (în secunde). Liniile întrerupte verticale negre marchează pozițiile stațiilor de sortare. Diferitele linii continue (împreună cu indicii autobuzelor) marchează cele patru autobuze non-rezervă. Linia punctată neagră reprezintă autobuzul de rezervă.

arrival_time(0)	arrival_time(50)
	

Constrângeri

- $1 \leq L \leq 10^9$
- $1 \leq N \leq 1\,000$
- $0 \leq T[i] \leq 10^{18}$ (pentru orice i cu $0 \leq i < N$)
- $1 \leq W[i] \leq 10^9$ (pentru orice i cu $0 \leq i < N$)
- $1 \leq X \leq 10^9$
- $2 \leq M \leq 1\,000$
- $0 = S[0] < S[1] < \cdots < S[M-1] = L$
- $1 \leq Q \leq 10^6$
- $0 \leq Y \leq 10^{18}$

Subprobleme

1. (9 puncte) $N = 1, Q \leq 1\,000$
2. (10 puncte) $M = 2, Q \leq 1\,000$
3. (20 de puncte) $N, M, Q \leq 100$
4. (26 de puncte) $Q \leq 5\,000$
5. (35 de puncte) Nicio constrângere suplimentară.

Exemplu de Grader

Exemplu de Grader citește de la tastatură date în următorul format:

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

Exemplu de Grader va printa răspunsurile voastre în următorul format:

- linie $1+k$ ($0 \leq k < Q$): valoarea returnată de `arrival_time` pentru întrebarea k