Adelantando

Hay una carretera de un solo carril desde el Aeropuerto de Budapest hasta el Hotel Forrás. La carretera es de L quilómetros de longitud.

Durante el evento de la IOI 2023, $N+1$ autobuses usan está carretera. Los buses están numerados de 0 a N. El bus i ($0 \leq i < N$) está programado para salir del aeropuerto en el segundo $T[i]$ del evento, y puede recorrer 1 quilómetro en $W[i]$ segundos. El bus N está reservado y puede recorrer 1 quilómetro en X segundos. El tiempo Y para salir del aeropuerto aún no se ha decidido.

Adelantar no está permitido en general, pero los autobuses tienen permitido adelantar en cada una de las **estaciones de ordenación**. Hay M ($M > 1$) estaciones de ordenación numeradas de 0 hasta $M-1$, en distintas posiciones de la carretera. La estación de ordenación j ($0 \leq j < M$) se encuentra $S[j]$ quilómetros desde el aeropuerto a lo largo de la carretera. Las estaciones de ordenación están ordenadas en orden creciente de la distancia al aeropuerto, es decir, $S[j] < S[j+1]$ para cada $0 \leq j \leq M-2$. La primera estación de ordenación es el aeropuerto y la última el hotel, es decir, $S[0] = 0$ y $S[M-1] = L$.

Cada bus viaja a su velocidad máxima, a menos que se encuentre un bus más lento delante en la carretera, en cuyo caso se agrupan, y ambos van a la velocidad del más lento llegar a una estación de ordenación. Allí, el autobús más rápido adelanta al autobús más lento.

Formalmente, para cada i y j tal que $0 \leq i \leq N$ y $0 \leq j < M$, el momento $t_{i,j}$ (en segundos) en que el bus i **llega a** la estación de ordenación j se define como sigue. Sea $t_{i,0} = T[i]$ para cada $0 \leq i < N$, y sea $t_{N,0} = Y$. Para cada j tal que $0 < j < M$:

- Definimos el **tiempo esperado de llegada** (en segundos) del bus i a la estación de ordenación j, denotado por $e_{i,j}$, como el momento en el que el bus i llegaría a la estación de ordenación j si viajase a velocidad máxima desde el tiempo que salió de la estación $j-1$. Esto es,
  - $e_{i,j} = t_{i,j-1} + W[i] \cdot (S[j] - S[j-1])$ para cada $0 \leq i < N$, y
  - $e_{N,j} = t_{N,j-1} + X \cdot (S[j] - S[j-1])$.
- El bus i llega a la estación de ordenación j en el *máximo* de los tiempos esperados de los buses i y cualquiera otro bus que hubiese llegado antes que el bus i a la estación $j-1$. Formalmente, sea $t_{i,j}$el máximo de $e_{i,j}$ y cada $e_{k,j}$ para el cual $0 \leq k \leq N$ y $t_{k,j-1} < t_{i,j-1}$.

Los organizadores de la IOI quieren programar el bus de reserva (bus N). Tu tarea es responder Q preguntas de los organizadores, que tienen la siguiente forma: dado el momento Y (en segundos) que se supone que el bus de reserva N sale del aeropuerto, ¿en qué momento llegará al hotel?

Detalles de implementación

Tu tarea es implementar los siguientes procedimientos.

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

- L: longitud de la carretera.
- N: número de autobuses no reservados.
- T: un array de longitud N describiendo los tiempos en los que los buses no reservados $0, \ldots, N-1$ están programados para salir del aeropuerto.
- W: un array de longitud N describiendo la velocidad máxima de los autobuses $0, \ldots, N-1$.
- X: el tiempo que tarda el autobús de reserva en recorrer 1 quilómetro.
- M: el número de estaciones de ordenación.
- S: un array de longitud M describiendo las distancias de las estaciones de ordenación hasta el aeropuerto
- Este procedimiento se llama exactamente una vez por cada juego de prueba, antes de cualquier llamada a `arrival_time`.

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

- Y: el tiempo en el que se supone que el autobús de reserva (bus N) sale del aeropuerto.
- Esta función debe devolver el momento en el que el autobús de reserva N llega al hotel.
- Esta función se llama exactamente Q veces.

Ejemplo

Considera la siguiente secuencia de llamadas:

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

Ignorando el bus 4 (que aún no se ha programado su salida), la siguiente tabla muestra los tiempos de llegada de los $0, 1, 2$ y 3 a cada estación de ordenación:

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

Los tiempos de llegada en la estación 0 son los tiempos en que los autobuses están programados para salir del aeropuerto. Eso es, $t_{i,0} = T[i]$ para $i = 0, 1, 2$ y 3.

Los tiempos esperados y los tiempos reales de llegada a la estación 1 se computan de la siguiente manera:

- Los tiempos esperados de 1:
  - 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: $e_{3,1} = t_{3,0} + W[3] \cdot (S[1] - S[0]) = 0 + 30 \cdot 1 = 30$.
- Los tiempos de llegada a1:
  - Buses 1 y 3 llegan a la estación 0 antes que el bus 0, por lo tanto $t_{0,1} = \max(e_{0,1}, e_{1,1}, e_{3,1}) = 30$.
  - Bus 3 llega a la estación 0 antes que el bus 1, por lo tanto $t_{1,1} = \max(e_{1,1}, e_{3,1}) = 30$.
  - Bus 0, bus 1 y bus 3 llegan a la estación de ordenación 0 antes que el bus 2, por lo tanto $t_{2,1} = \max(e_{0,1}, e_{1,1}, e_{2,1}, e_{3,1}) = 60$.
  - Ningún bus llega a la estación 0 antes que el bus 3, por lo tanto $t_{3,1} = \max(e_{3,1}) = 30$.

```
arrival_time(0)
```

El bus 4 necesita 10 segundos para viajar 1 quilímetro y está programado para salir del aeropuerto en el segundo 0. En este caso, la siguiente tabla muestra los tiempos de llegada de cada bus. Los cambios respecto a la tabla inicial están subrallados.

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

Podemos ver que el bus 4 llega al hotel en el segundo 60. Así pues, la función debe devolver 60.

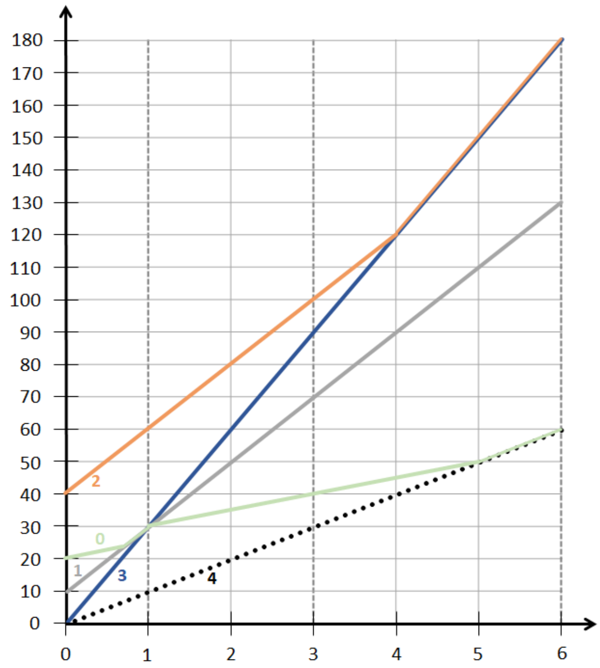
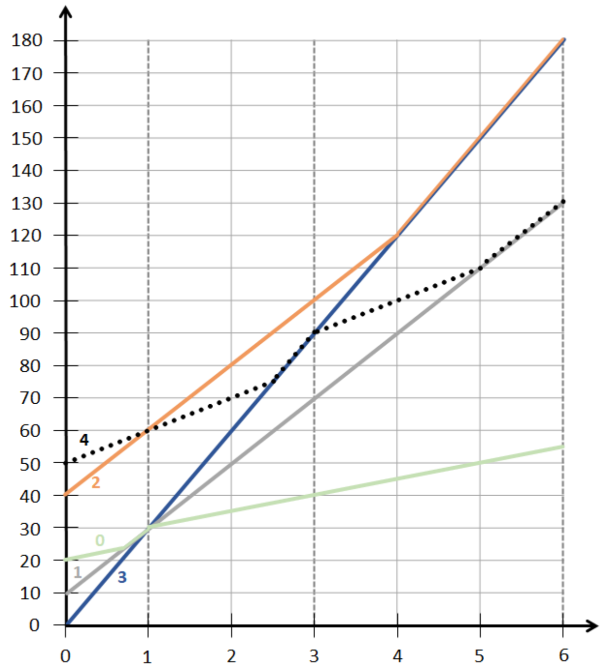
```
arrival_time(50)
```

Bus 4 está programado para salir del aeropuerto en el segundo 50. En este caso no hay cambios para los tiempos de llegada de los autobuses 0, 1, 2 y 3. Los tiempos de llegada se muestran en la siguiente tabla:

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

El bus 4 adelanta al bus lento 2 en la estación de ordenación 1 al llegar al mismo tiempo. Después, bus 4 se agrupa con el bus 3 entre las estaciones 1 y la 2, haciendo que el bus 4 llegue a la estación 2 en el segundo 90 en vez del 80. Después de dejar la estación 2, bus 4 se agrupa al bus 1 hasta que llegan al hotel. El autobús 4 llega al hotel en el segundo 130. Así pues, tu función debe devolver 130.

Podemos hacer una gráfica del tiempo que necesita cada autobús para llegar a cada distancia del aeropuerto. El eje de las x representa la distancia del aeropuerto (en quilómetros) y el eje de las y representa el tiempo (en segundos). Las líneas verticales representan las estaciones de ordenación. Las diferentes líneas sólidas (acompañadas del índice del bus) representan los cuatros autobuses no reservados (no el de reserva). La línea negra de puntos representa el autobús de reserva.

`arrival_time(0)`	`arrival_time(50)`
	

Restricciones

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

Subtareas

1. (9 puntos) $N = 1, Q \leq 1\,000$
2. (10 puntos) $M = 2, Q \leq 1\,000$
3. (20 puntos) $N, M, Q \leq 100$
4. (26 puntos) $Q \leq 5\,000$
5. (35 puntos) Sin restricciones adicionales.

Grader de ejemplo

El grader de ejemplo lee la entrada en el siguiente formato:

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

El grader de ejemplo escribe tu respuesta en el siguiente formato:

- líneas $1+k$ ($0 \leq k < Q$): el valor devuelto por `arrival_time` para la pregunta k