Tracé de montagne russe

Anna travaille dans un parc d'attraction et est en charge de construire le tracé d'une nouvelle montagne russe. Elle a déjà conçu n sections (numérotées de 0 à $n-1$) qui affectent la vitesse du train. Elle doit maintenant les assembler et proposer le tracé final de montagne russe. Pour ce problème, vous pouvez considérer que la longueur du train est nulle.

Pour chaque i compris entre 0 et $n-1$ (inclus), la section i a deux propriétés :

- à l'entrée de la section, une vitesse limite : la vitesse du train doit être **plus petite ou égale** à s_i km/h,
- à la sortie de la section, la vitesse du train est d'**exactement** t_i km/h, quelle que soit la vitesse d'entrée du train dans la section.

La montagne russe terminée est un tracé qui contient n sections dans un ordre particulier, chaque section étant utilisée exactement une fois. Les sections consécutives sont reliées par des connexions. Anna doit choisir l'ordre des n sections et ensuite décider de la longueur de chaque connexion. La longueur d'une connexion est mesurée en mètres et peut valoir tout entier positif ou nul.

Chaque mètre de connexion entre deux sections ralentit le train de 1 km/h. Au début de l'attraction, le train entre dans la première section choisie par Anna à 1 km/h.

Le tracé final doit satisfaire les contraintes suivantes :

- le train ne dépasse pas les vitesses limites en entrant dans les sections ;
- la vitesse du train est positive à tout moment.

Dans toutes les sous-tâches sauf dans la sous-tâche 3, vous devez trouver la longueur minimale totale possible des connexions entre les sections. Dans la sous-tâche 3, vous devez uniquement vérifier s'il est possible de concevoir une montagne russe telle que chaque connexion a une longueur nulle.

Détails d'implémentation

Vous devez implémenter la fonction (méthode) suivante :

- `plan_roller_coaster(int[] s, int[] t)`.
  - `s`: tableau de longueur n, vitesses maximales d'entrée.
  - `t`: tableau de longueur n, vitesses de sortie.
  - Dans toutes les sous-tâches excepté la 3, la fonction doit retourner la longueur totale minimale des connexions. Dans la sous-tâche 3, vous devez retourner 0 s'il existe un tracé de montagne russe valide de telle façon à ce

que chaque connexion ait une longueur nulle, et n'importe quel entier strictement positif s'il n'en existe pas.

Pour le langage C, la signature de la fonction est légèrement différente :

- `int64 plan_roller_coaster(int n, int[] s, int[] t)`
  - `n`: le nombre d'éléments dans `s` et `t` (c-à-d., le nombre de sections),
  - les autres paramètres sont les mêmes que ci-dessus.

Exemple

`int64 plan_roller_coaster([1, 4, 5, 6], [7, 3, 8, 6])`

Dans cet exemple, il y a quatre sections. La meilleure solution est de les construire dans l'ordre $0, 3, 1, 2$ et de les relier avec des connexions de longueurs respectives $1, 2, 0$. Le train parcourt le trajet comme suit :

- Initialement la vitesse du train est de 1 km/h.
- Le train commence son parcours en entrant dans la section 0.
- Le train quitte la section 0 à une vitesse de 7 km/h.
- Ensuite il y a une connexion de longueur de 1 m. Lorsque le train atteint la fin de la connexion, sa vitesse est de 6 km/h.
- Le train entre dans la section 3 à 6 km/h et la quitte à la même vitesse.
- Après avoir quitté la section 3, le train parcourt une connexion de 2 m de long. Sa vitesse descend à 4 km/h.
- Le train entre dans la section 1 à 4 km/h et la quitte à 3 km/h.
- Immédiatement après la section 1 le train entre dans la section 2.
- Le train quitte la section 2. Sa vitesse finale est de 8 km/h.

La fonction doit retourner la longueur totale des connexions : $1 + 2 + 0 = 3$.

Sous-tâches

Dans chaque sous-tâche, $1 \leq s_i \leq 10^9$ et $1 \leq t_i \leq 10^9$.

1. (11 points): $2 \leq n \leq 8$,
2. (23 points): $2 \leq n \leq 16$,
3. (30 points): $2 \leq n \leq 200\,000$. Dans cette sous-tâche, votre programme doit uniquement vérifier si la réponse est zéro ou non. Si votre réponse n'est pas zéro, tout nombre entier strictement positif est considéré correct.
4. (36 points): $2 \leq n \leq 200\,000$.

Évaluateur fourni (grader)

L'évaluateur fourni lit l'entrée dans le format suivant :

- ligne 1 : l'entier n.
- ligne 2 + i, pour i de 0 à $n - 1$: les entiers s_i et t_i.