The last of us

1 le background

En cas de mutinerie, désertion, faute grave de la part de légionnaires, l'armée romaine utilisait un procédé extrêmement violent pour maintenir la terreur : la décimation. Dix inculpés devaient tirer au sort lequel serait exécuté par les 9 autres. D'où l'orignie du mot décimation. Les boureaux étaient bien entendu punis également, mais sans mourir.

Actuellement, décimation n'est plus vraiment relié à son sens strict puisque l'on parle de décimation quand bien même le rapport morts-vivants dépasse 1 sur 10. On emploie même ce terme lorsqu'il ne reste plus rien du tout. C'est pourquoi il est utilisé pour qualifier certains types d'algorithmes informatique, dont le but est de faire une sélection parmi un ensemble de données (par exemple un tableau) pour ne garder qu'un ensemble réduit.

2 l'énoncé

L'objectif de l'exercice est d'appliquer une stratégie de décimation simple sur un ensemble de données, qui n'en conserve qu'une seule à la fin et de déterminer son indice. Pour décrire cette décimation, on utilise une analogie. On suppose qu'il y a une file $N \geq 2$ personnes, chacune étant numérotée de 1 à M. On commence par éliminer celle qui a le numéro $D \geq 1$. Ensuite, en direction de la fin de la file, on élimine celle qui se trouve $S \geq 1$ places plus loin. Si on atteint la fin de file avant d'avoir compté S vivants, on retourne au début. On repète ce processus jusqu'à ce qu'il ne reste plus qu'une seule personne vivante. On connaît alors son numéro.

Exemple avec $N = 4$, $D = 3$ et $S = 2$:

1. on constitue la file avec `1 2 3 4`.
2. on élimine la 3ème personne de la file (car $D = 3$). La file restante contient `1 2 4`.
3. à partir de la place de la personne qui vient d'être éliminée, on compte 2 vivants (car $S = 2$). Le premier est le n° 4 et comme on atteint la fin de file, on revient au début, pour trouver le deuxième vivant, à savoir le n° 1, que l'on élimine. La file restante contient `2 4`.
4. on compte 2 vivants, donc le n°2 puis le n°4. Ce dernier est éliminé.
5. le n° du survivant est donc 2.

Pour résoudre l'exercice, votre programme doit lire sur l'entrée standard :

1. un entier M représentant le nombre de triplets à traiter.

2. M chaînes de caractères au format `N D S`, sachant que `N` est le nombre de personnes, `D` le n° de celle qui est éliminée en premier, et `S` le pas.

Votre programme doit écrire sur la sortie standard M lignes. La i^{eme} ligne contient le n° du survivant pour le i^{eme} triplet. valeur.

Exemple d'entrée/sortie :

entrée	sortie
6	2
2 1 1	1
2 2 1	2
4 3 2	3
5 2 3	7
7 1 1	4
4 7 5	

TABLE 1 – Exemple d'entrée et la sortie attendue

IMPORTANT :

— comme on peut le constater avec la dernière ligne de l'exemple ci-dessus, `D` peut être plus grand que `N`. Dans ce cas, il suffit de faire comme avec `S`, à savoir revenir au début de la file. Dans le cas présent, éliminer le 7ème sur 4, revient à éliminer le 3ème.

— le temps maximal pour calculer la solution des M triplets est limité à 2 secondes. Si votre programme n'est pas capable de calculer la solution du triplet `6666 999 9999`, à savoir `1289`, en moins de 2 secondes, il ne sera pas capable non plus de calculer les solutions de certains fichiers secret en moins de 2 secondes.

3 les ressources

Pour vous aider dans la réalisation du programme, vous trouverez sur
`http://cours-info.iut-bm.univ-fcomte.fr`
un article dans la section `hackaton` de l'année courante, portant le même titre que l'exercice. Il contient un lien permettant de télécharger un canevas de code, ainsi que le fichier d'entrée donné ci-dessus.

Bien entendu, vous êtes libres d'utiliser ou non ce canevas, mais c'est un gain de temps que de s'en servir comme base.