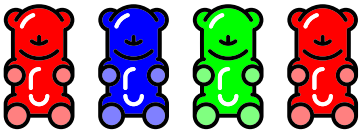
Bonbons oursons

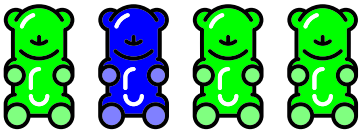
La souris Stofl adore les bonbons en forme d'oursons et les mange d'une façon particulière. Un paquet d'oursons contient n bonbons. Chaque bonbon a un certain goût, donné comme un nombre entier.

Stofl veut manger le paquet en prenant plusieurs fois des bouchées de k bonbons (il est garanti que k est un diviseur de n). Une bouchée de bonbons est *délicieuse* s'il y a un goût majoritaire, c'est-à-dire s'il y a un certain goût qui est celui de plus de $k/2$ bonbons de ladite bouchée. S'il n'y a pas de goût majoritaire, Stofl trouve cette bouchée *répugnante*.

Par exemple, si $k = 4$, une bouchée doit contenir strictement plus de $k/2 = 2$, donc au moins 3 goûts de la même sorte.



Berk ! Il n'y a pas de goût majoritaire.



Miam ! Le vert est le goût majoritaire.

Est-il possible pour Stofl de manger les bonbons de façon à ce que chaque bouchée soit délicieuse ? Si oui, imprime une façon possible de les manger.

Entrée

La première ligne de l'entrée contient trois entiers n, k et p – le nombre de bonbons, n, la taille de chaque bouchée, k, et si vous devez aussi imprimer comment le manger (si $p = 0$ vous n'avez pas à l'imprimer, si $p = 1$ alors vous le devez).

La deuxième ligne contient n entiers f_i – les goûts des bonbons.

Sortie

S'il est possible de manger tous les bonbons de façon à ce que chaque bouchée soit délicieuse, imprime "`Yummy!`" sur la première ligne, sinon "`Yuck!`".

Selon la sous-tâche, c'est-à-dire si $p = 1$, et si la première ligne était "`Yummy!`", tu dois aussi imprimer une façon de les manger. Pour cela, imprime n/k lignes, contenant chacune k nombres entiers : la i-e de ces lignes doit contenir les goûts que tu veux mettre dans la i-e bouchée. Tu peux utiliser chaque goût autant de fois qu'il apparaît dans l'entrée.

Quand $p = 0$, chaque ligne à part la première est ignorée.

S'il y a plusieurs possibilités, tu peux imprimer n'importe laquelle d'entre elles.

Limites

Il y a huit sous-tâches. Dans chaque sous-tâche, $1 \leq k \leq n$.

Pour les sous-tâches impaires (1, 3, 5 et 7), on a $p = 0$.

Pour les sous-tâches paires (2, 4, 6 et 8), nous avons $0 \leq p \leq 1$.

Dans les exemples, $p = 1$. Si vous prévoyez de résoudre uniquement les sous-tâches avec avec $p = 0$, vous pouvez toujours voir dans le verdict si la première ligne était correcte.

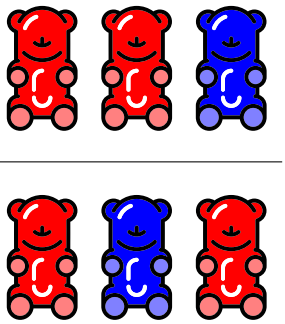
- Dans les sous-tâches 1 et 2, valant chacune 10 points : $n \leq 100\,000$, $0 \leq f_i < n/k$ et chaque goût apparaît exactement k fois. En d'autres termes, les f_i contiennent k fois 0, k fois 1, etc., et k fois $n/k - 1$.
- Dans les sous-tâches 3 et 4, valant chacune 10 points : $n \leq 100\,000$ et $0 \leq f_i < 2$, i.e. il y a au plus deux goûts différents.

— Dans les sous-tâches 5 et 6, valant chacune 10 points : $n \leq 1000$ et $0 \leq f_i < n$.
— Dans les sous-tâches 7 et 8, valant chacune 20 points : $n \leq 100\,000$ et $0 \leq f_i < 10^9$.

Exemples

Entrée	Sortie
6 3 1 1 0 0 0 0 1	Yummy! 0 0 1 0 1 0

Cette entrée serait valide pour les sous-tâches 3, 4, 5, 6, 7 et 8. La solution ressemble à ceci (avec rouge = 0 et bleu = 1) :



Entrée	Sortie
16 4 1 0 0 0 1 1 1 1 1 1 1 1 3 3 4 4	Yummy! 0 0 0 3 1 1 1 3 1 1 1 4 1 1 1 4

Cette entrée serait valide pour les sous-tâches 5, 6, 7 and 8.

Entrée	Sortie
12 3 1 0 0 0 0 0 1 2 2 2 3 4 5	Yuck!

Cette entrée serait valide pour les sous-tâches 5, 6, 7 and 8.

Entrée	Sortie
4 2 1 0 1 0 0	Yuck!

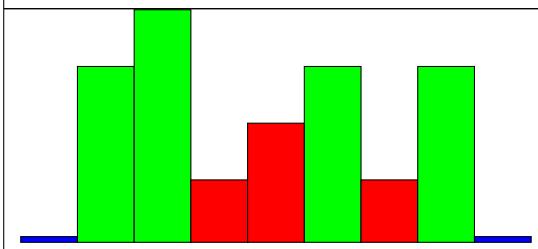
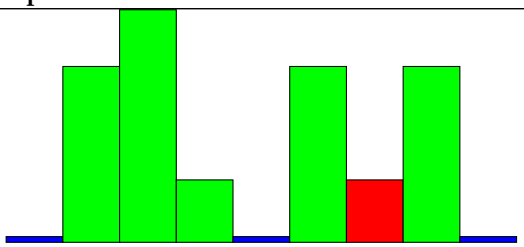
Cette entrée serait valide pour les sous-tâches 3, 4, 5, 6, 7 and 8.

Entrée	Sortie
12 4 1 0 1 2 2 1 1 1 0 2 2 0 0	Yummy! 0 0 0 0 1 1 1 1 2 2 2 2

Cette entrée serait valide pour toutes les sous-tâches.

Vue d'eau

La souris Stofl construit un chemin de randonnée entre deux lacs. Le chemin consiste de n segments, où chaque segment i est à hauteur h_i. Les deux lacs sont à hauteur 0, et ils sont positionnés avant le 0-ième et après le $(n-1)$-ième segment respectivement. Stofl veut construire le plus beau chemin possible. Un segment i du chemin est *beau* si on peut voir un lac, c'est-à-dire s'il y a un lac tel qu'il n'y a aucun segment plus haut que h_i entre le segment i et ce lac. La beauté de la randonnée est le nombre de beaux segments qu'il contient. Stofl peut remplacer exactement un segment par un lac supplémentaire, aussi à hauteur 0. Ce lac sera aussi considéré comme un segment beau. (Pour que les randonneurs ne doivent pas nager, Stofl construit un pont d'où l'on peut bien sûr voir le lac.)

Avant	**Après**
	
C'est la situation avant la construction du lac supplémentaire. Trois des segments n'ont pas de vue sur l'eau, donc la beauté est 4.	Après avoir construit un nouveau lac au milieu, tout sauf un segment deviennent beaux et la nouvelle beauté est 6.

Entrée

La première ligne de l'entrée contient un entier n, le nombre de segments. La prochaine ligne contient n entiers h_i, les hauteurs des segments.

Sortie

Imprime un seul entier, le nombre maximal de beaux segments que Stofl peut atteindre.

Limites

Dans tous les cas de test $1 \leq h_i \leq 10^9$ et $1 \leq n \leq 500\,000$.

— Dans le premier groupe de test qui vaut 30 points, $1 \leq n \leq 500$.
— Dans le deuxième groupe de test qui vaut également 30 points, $1 \leq n \leq 10\,000$.
— Dans le troisième groupe de test qui vaut 40 points, il n'y a pas de restrictions supplémentaires.

Exemples

Entrée	Sortie
3 2 1 3	3

Stofl peut remplacer soit l'un des trois segments avec un lac afin de rendre beau tous les segments.

Entrée	Sortie
7 3 4 1 2 1 3 2	7

Le segment avec hauteur 2 est le seul segment que Stofl peut convertir afin de rendre beau tous les segments.

Entrée	Sortie
7 3 4 1 2 3 1 3	6

Stofl ne peut pas rendre beau tous les segments. S'il converti le segment de hauteur 2, six segments seront beaux.

Glisse

La souris Daniel se tient au milieu d'une patinoire. Celle-là est représentée par une grille rectangulaire de dimensions $n \times m$, consistant en trois types de cases :

— Des cases Mur "`#`" : Daniel ne peut pas se déplacer sur ces cases.
— Des cases Glace "`+`" : En se déplaçant sur une case de ce type, Daniel glisse dessus, donc il part de la case dans la même direction (sauf s'il y a un mur dans cette direction, auquel cas il s'arrête).
— Des cases Terre "`.`" : En arrivant sur une case de ce type, Daniel s'arrête. Il peut partir de la case dans n'importe laquelle des quatre directions.

Il y a deux cases Terre spéciales :

— La case de départ "`d`" : Daniel commence ici.
— La case d'arrivée "`g`" : Daniel gagne s'il atteint cette case.

Daniel se déplace à une vitesse d'une case par seconde. Peux-tu l'aider à atteindre la case d'arrivée le plus vite possible ?

Entrée

La première ligne contient deux entiers, n et m. Après cela, m lignes suivent, contenant chacune n caractères. Chaque caractère est l'un des suivants : `#`, `+`, `.`, `d`, `g`. `d` et `g` n'ont qu'une occurrence chacun. Toutes les cases du bord sont de type `#`.

Sortie

Imprime le nombre de secondes dont Daniel a besoin pour atteindre l'arrivée, ou "`IMPOSSIBLE`" si c'est impossible.

Limites

Il y a 4 groupes de test, valant chacun 25 points.

— Dans le groupe 1, $n, m \leq 100$ et il n'y a pas de case Glace.
— Dans le groupe 2, $n, m \leq 100$.
— Dans le groupe 3, $n, m \leq 1000$.
— Dans le groupe 4, $n, m \leq 2000$.

Exemples

Entrée	Sortie
8 5 ######## #.++g#.# #.+++.d# #..+..+# ########	5

Daniel ne peut pas aller vers le haut puis à gauche, car il y a un mur sur le chemin. S'il marche deux cases à gauche, il glisse sur la glace jusqu'au mur de gauche, et cela lui prend 9 secondes d'atteindre l'arrivée. Un trajet plus rapide est de commencer par aller vers le bas (Daniel sera arrêté par le mur), puis deux cases vers la gauche, et ensuite il peut glisser sur la glace jusqu'à l'arrivée.

Entrée	Sortie
5 3 ##### #d#g# #####	IMPOSSIBLE

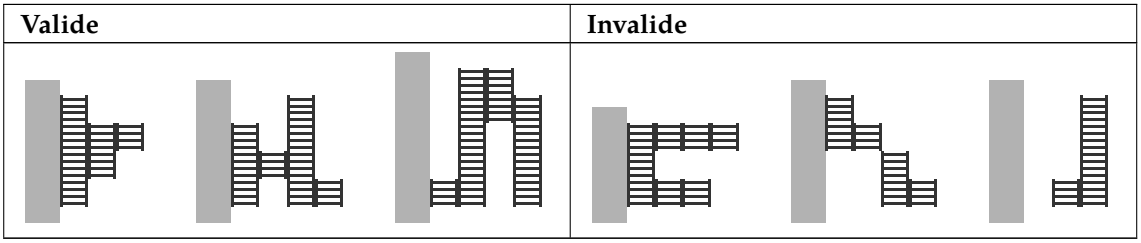
Malheureusement, la case d'arrivée est derrière un mur et ne peut pas être atteinte.

Balcon

La souris Binna vient de finir de construire une énorme tour et aimerait y construire un balcon. Elle ne veut pas juste un ennuyeux balcon ordinaire, non, le balcon doit être fait d'échelles. Cela lui permettrait de monter et descendre tout en appréciant le soleil.

Si on imagine la tour à gauche et le coucher de soleil devant et à droite, alors les échelles peuvent être installées comme suit : l'échelle la plus à gauche doit être fixée à la tour et chaque autre échelle doit être fixée à l'échelle directement à sa gauche. Pour que deux échelles soient fixées ensemble correctement, elles doivent partger une arête, pas juste un angle.

En outre, il ne peut jamais y avoir deux échelles l'une en dessus de l'autre avec un trou entre les deux, car Binna pourrait tomber et se briser les jambes si elle ne voit pas le trou.



La tour fait H mètres de haut et Binna estime que la tour peut soutenir des échelles allant jusqu'à W mètres plus loin. elle a déjà acheté une échelle qui fait N mètres de long et 1 de large. Elle peut couper cette échelle en morceaux, mais pour des raisons esthétiques, chacune de ces pièces doit avoir une longueur qui soit un nombre entier. Ensuite, elle fixe les pièces ensemble pour construire son balcon.

En utilisant un drone, Binna a déjà calculé combien le soleil envoie de lumière sur la surface de $H \times W$ mètres devant sa tour. Elle aimerait maximiser la quantité totale de lumière du soleil que son balcon reçoit. Peux-tu l'aider à le faire ?

Entrée

La première ligne de l'entrée contient trois entiers H, W et N – la hauteur de la tour, la distance maximale sur laquelle le balcon s'étend et le nombre de mètres d'échelle à disposition de Binna. Les H lignes suivantes contiennent chacune W nombres entiers $a_{i,j}$ qui décrivent la quantité de lumière dans la zone devant sa tour.

Sortie

Imprime un unique entier – le total maximal de lumière du soleil que peut recevoir le balcon de Binna.

Limites

Dans tous les cas de test, $1 \leq H, W, N$ et $N \leq H \cdot W$ et $0 \leq a_{i,j} \leq 10^6$.

— Dans la première sous-tâche (20 points), $H, W, N \leq 20$.
— Dans la deuxième sous-tâche (20 points), $H, W, N \leq 40$.
— Dans la troisième sous-tâche (20 points), $H, W, N \leq 80$.
— Dans la quatrième sous-tâche (20 points), $H \leq 100$, $W \leq 80$ et $N \leq 100$.
— Dans la cinquième sous-tâche (10 points), $H \leq 120$, $W \leq 50$ et $N \leq 120$.
— Dans la sixième sous-tâche (10 points), $H \leq 130$, $W \leq 30$ et $N \leq 130$.

Exemples

Entrée	Sortie
2 3 3 1 2 3 7 8 9	24

Entrée	Sortie
7 6 10 2 6 1 1 2 9 6 2 9 1 9 3 7 5 6 8 2 6 5 6 5 4 5 4 6 7 3 6 3 8 4 1 4 5 2 1 3 2 5 4 6 3	65