Statement is not available on English language

A. Tuyauterie

1 second, 256 megabytes

En te lavant les mains, tu t'es rendu compte que ton évier, si ouvert au maximum, débordait. Quand tu as appelé ton plombier, quelle ne fût pas sa surprise quand il vit la tuyauterie.

L'évier consiste en n tuyaux directement connectés à l'évier et à chacun de ses tuyaux m - 1 tuyaux sont accrochés. Chaque tuyau à une capacité différente. Le plombier à calculer qu'il était trop compliqué de réparer la tuyauterie, mais qu'il suffisait de ne pas ouvrir le robinet à capacité plus que x. Malheureusement, tu ne l'écoutais pas quand il a annoncé la valeur de x. Pour éviter tout problème, tu dois retrouver la valeur de x en regardant la composition de ta tuyauterie.

Input
La première ligne de l'entrée consiste en deux entiers n et m $(1 \le n, m \le 1000)$ – le nombre de chaines de tuyaux et la longueur de chaque chaine.

Puis, il y a n lignes avec m entiers a_1, a_2, ..., a_m la capacité de chaque tuyau $(1 \le a_i \le 1000)$

Output
Il faut imprimer une ligne, la valeur de x.

input
2 3 4 6 5 1 3 5
output
5

B. Grands Graphes

2 s., 256 MB

Le fermier John possède une ferme composée de n pâturages reliés par des routes unidirectionnelles. Chaque route a un poids, représentant le temps nécessaire pour aller du début à la fin de la route. Les routes pourraient avoir un poids négatif, où les vaches vont si vite qu'elles retournent dans le temps ! Cependant, le fermier John garantit qu'il est impossible que les vaches restent coincées dans une boucle temporelle, où elles peuvent voyager indéfiniment dans le temps en traversant une séquence de routes. De plus, chaque paire de pâturages est reliée par au plus une route dans chaque direction.

Malheureusement, le fermier John a perdu la carte de la ferme. Tout ce dont il se souvient est un tableau d, où d_i est la plus petite quantité de temps qu'il a fallu aux vaches pour atteindre le i-ème pâturage depuis le pâturage 1 en utilisant une séquence de routes. Le coût de sa ferme est la somme des poids de chacune des routes, et le fermier John a besoin de connaître le coût **minimal** d'une ferme conforme à sa mémoire.

Input
La première ligne contient un entier t ($1 \le t \le 10^4$) — le nombre de cas de test. Ensuite, t cas suivent.

La première ligne de chaque cas de test contient un seul entier n ($1 \le n \le 10^5$) — le nombre de pâturages.

La deuxième ligne de chaque cas de test contient n entiers séparés par des espaces $d_1, d_2, \ldots, d_n$ ($0 \le d_i \le 10^9$) — le tableau d. Il est garanti que $d_1 = 0$.

Il est garanti que la somme de n sur tous les cas de test ne dépasse pas 10^5.

Output
Pour chaque cas de test, imprimez le coût minimum possible d'une ferme conforme à la mémoire du fermier John.

input
3 3 0 2 3 2 0 1000000000 1 0
output
-3 0 0

Dans le premier cas de test, vous pouvez ajouter des routes

- du pâturage 1 au pâturage 2 avec un temps de 2,
- du pâturage 2 au pâturage 3 avec un temps de 1,
- du pâturage 3 au pâturage 1 avec un temps de -3,
- du pâturage 3 au pâturage 2 avec un temps de -1,
- du pâturage 2 au pâturage 1 avec un temps de -2.

Le coût total est $2 + 1 + -3 + -1 + -2 = -3$.
Dans le deuxième cas de test, vous pouvez ajouter une route du pâturage 1 au pâturage 2 avec un coût de 1000000000 et une route du pâturage 2 au pâturage 1 avec un coût de -1000000000. Le coût total est $1000000000 + -1000000000 = 0$.

Dans le troisième cas de test, vous ne pouvez pas ajouter de routes. Le coût total est 0.

C. Chemin fléché

2 s., 256 MB

Il y a une grille, composée de 2 lignes et n colonnes. Les lignes sont numérotées de 1 à 2 de haut en bas. Les colonnes sont numérotées de 1 à n de gauche à droite. Chaque cellule de la grille contient une flèche pointant soit vers la gauche soit vers la droite. Aucune flèche ne pointe à l'extérieur de la grille.

Il y a un robot qui démarre dans une cellule $(1, 1)$. À chaque seconde, les deux actions suivantes se produisent successivement :

1. Tout d'abord, le robot se déplace vers la gauche, la droite, vers le bas ou vers le haut (**il ne peut pas essayer de sortir de la grille, et ne peut pas sauter un mouvement**);
2. ensuite, il se déplace le long de la flèche placée dans la cellule actuelle (la cellule dans laquelle il se retrouve après son déplacement).

Votre tâche est de déterminer si le robot peut atteindre la cellule $(2, n)$.

Input
La première ligne contient un seul entier t ($1 \le t \le 10^4$) — le nombre de cas de test.

La première ligne de chaque cas de test contient un seul entier ($2 \le n \le 2 \cdot 10^5$).

La deuxième ligne contient une chaîne de caractères comprenant exactement n caractères < et/ou > — la première ligne de la grille.

La troisième ligne contient une chaîne de caractères comprenant exactement n caractères < et/ou > — la deuxième ligne de la grille.

Contraintes supplémentaires sur l'entrée :

- n est pair ;
- il n'y a pas de flèches pointant à l'extérieur de la grille ;
- la somme de n sur tous les cas de test ne dépasse pas $2 \cdot 10^5$.

Output
Pour chaque cas de test, imprimez `YES` si le robot peut atteindre la cellule $(2, n)$; sinon, imprimez `NO`.

Vous pouvez imprimer chaque lettre dans n'importe quel cas. Par exemple, `yes`, `Yes`, `YeS` seront tous reconnus comme une réponse positive.

input
4 4 >><< >>>< 2 > > 4 >>>< >><< 6 >><<>< ><>>><

output
YES YES NO YES

Dans le premier exemple, l'un des chemins possibles est le suivant : $(1, 1) \to (1, 2) \to (1, 3) \to (2, 3) \to (2, 4)$.

Dans le deuxième exemple, l'un des chemins possibles est le suivant : $(1, 1) \to (2, 1) \to (2, 2)$.

Dans le troisième exemple, il n'y a aucun moyen d'atteindre la cellule $(2, 4)$.

Dans le quatrième exemple, l'un des chemins possibles est le suivant : $(1, 1) \to (2, 1) \to (2, 2) \to (1, 2) \to (1, 3) \to (2, 3) \to (2, 4) \to (2, 5) \to (2, 6)$.