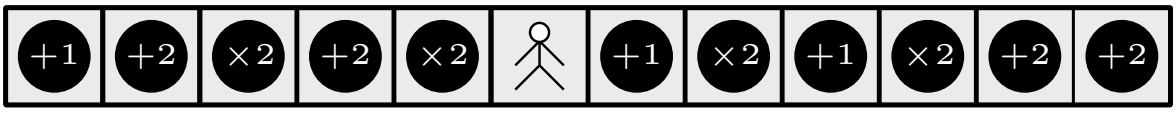
COURS 2 : ALGORITHMIQUE AVANCÉE

PROBLÈME 1 – Trouver le highscore

Dans un jeu vidéo, le personnage jouable se trouve coincé dans un couloir rempli de bonus. Ces bonus sont de 3 types différents : $+1$, $+2$, $\times 2$. Ils ont pour effet d'incrémenter le score, d'augmenter le score par 2, de multiplier le score par 2 respectivement.



Le personnage ne peut se déplacer que vers la gauche et vers la droite, et quand il marche sur un bonus, il est obligé de le prendre.

Quel est le score maximal que vous pouvez obtenir ?

Entrée

La première ligne contient un seul entier t ($1 \leq t \leq 100$) — le nombre de tests.
La première ligne de chaque test contient 1 entiers n ($1 \leq n \leq 20000$) – la taille du tableau. La deuxième de chaque test est la succession de n chaînes de caractères parmi `"+1"`, `"+2"`, `"x2"` et `"O"`. La chaîne de caractère `"O"` – représentant la position du départ du personnage – apparaît forcément une unique fois dans le tableau.

Sortie

Pour chaque test, affichez le score maximal que peut atteindre le joueur en ramassant tous les bonus.

Exemple

Entrée :

```
2
12
+1 +2 x2 +2 x2 O +1 x2 +1 x2 +2 +2
13
+2 +2 x2 +2 x2 x2 O x2 +1 x2 +2 x2 +1
```

Sortie :

```
50
81
```

Pour atteindre le score de `50` pour le premier test, il faut récupérer le +1 à droite, puis tous les bonus à gauche et enfin tous les bonus à droite restant. Le score est alors égal à

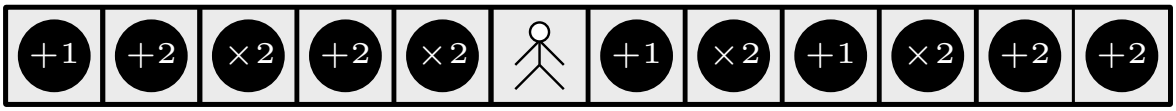
$$0 \to_{+1} 1 \to_{\times 2} 2 \to_{+2} 4 \to_{\times 2} 8 \to_{+2} 10 \to_{+1} 11 \to_{\times 2} 22 \to_{+1} 23 \to_{\times 2} 46 \to_{+2} 48 \to_{+2} 50.$$

Pour le deuxième, il suffit de ramasser les bonus dans l'ordre :

$$0 \to_{\times 2} 0 \to_{+1} 1 \to_{\times 2} 2 \to_{+2} 4 \to_{\times 2} 8 \to_{\times 2} 16 \to_{+2} 18 \to_{\times 2} 36 \to_{+2} 38 \to_{+2} 40 \to_{\times 2} 80 \to_{+1} 81.$$

PROBLÈME 2 – Déplacements optimaux

On reprend le problème précédent mais cette fois, il faut dire quels sont les mouvements qui ont permis d'atteindre le meilleur score.



Entrée

La première ligne contient un seul entier t ($1 \leq t \leq 100$) — le nombre de tests.
La première ligne de chaque test contient 1 entiers n ($1 \leq n \leq 20000$) – la taille du tableau. La deuxième de chaque test est la succession de n chaînes de caractères parmi `"+1"`, `"+2"`, `"x2"` et `"O"`. La chaîne de caractère `"O"` – représentant la position du départ du personnage – apparaît forcément une unique fois dans le tableau.

Sortie

Pour chaque test, affichez les déplacements du personnage à effectuer pour avoir le meilleur score.

Le format de réponse attendu doit être de la forme `entier1 lettre1 entier2 lettre2...` où les lettres valent `G` ou `D` (pour Gauche et Droite) et les entiers correspondent au nombre de mouvement que le personnage doit faire dans cette direction. Par exemple `2 G 3 D 1 G` signifie que le personnage va d'abord à gauche 2 fois, puis à droite 3 fois et enfin 1 fois à gauche.

Si plusieurs solutions sont possibles, affichez n'importe laquelle.

Exemple

Entrée :

```
2
12
+1 +2 x2 +2 x2 O +1 x2 +1 x2 +2 +2
13
+2 +2 x2 +2 x2 x2 O x2 +1 x2 +2 x2 +1
```

Sortie :

```
1 D 6 G 11 D
4 D 10 G 12 D
```

Pour atteindre le score de `50` pour le premier test, on fait 1 mouvement à droite (on est alors sur la septième case), puis 6 mouvements à gauche (on arrive sur la première case) et enfin 11 mouvements à droite pour récupérer le bonus tout à droite.

On aurait aussi pu afficher `1 D 2 G 1 G 1 G 1 G 1 G 11 D`.

PROBLÈME 3 – Chemin le plus long (S'il y a le temps...)

On nous donne les arêtes d'un graphe orientés dont les sommets sont représentés par des entiers (cf l'exemple ci-contre, la première ligne signifie qu'il y a une arête qui relie `0` à `2` dans cet ordre). **On suppose que le graphe est acyclique** (i.e. pas possible de faire un chemin dont le sommet de départ et le sommet d'arrivée).

Écrire un algorithme qui trouve un des chemins les plus longs reliant `0` à `1`.

Sur l'exemple ci-contre, on devra afficher `0 3 2 1`.

Entrée :

```
0 2
0 3
0 4
2 1
3 1
3 2
4 1
```