carbon IT

Exercice pratique - La carte aux trésors

Guidez les aventuriers en quête de trésors !

Contexte

Le gouvernement péruvien vient d'autoriser les aventuriers en quête de trésors à explorer les 85 182 km² du département de la Madre de Dios. Vous devez réaliser un système permettant de suivre les déplacements et les collectes de trésors effectuées par les aventuriers. Le gouvernement péruvien étant très à cheval sur les bonnes pratiques de code, il est important de réaliser un code de qualité, lisible, et maintenable (**oui, ça veut dire avec des tests**) !

Données du problème

La carte

La carte de la Madre de Dios est de forme rectangulaire, chaque case ayant la même taille. On y trouve des plaines, des montagnes et des trésors.

Les dimensions de la carte sont définies dans le fichier d'entrée de l'exercice par la ligne suivante :

```
# {C comme Carte} - {Nb. de case en largeur} - {Nb. de case en hauteur}
C - 3 - 4
```

Par défaut, toutes les cases de la carte sont des plaines que les aventuriers peuvent traverser sans encombre. Les cases sont numérotées d'ouest en est, de nord en sud, en commençant par zéro.

Les montagnes sont des obstacles infranchissables pour les aventuriers. Chaque montagne de la carte de la Madre de Dios est également indiquée dans le fichier d'entrée de l'exercice par la ligne suivante :

```
# {M comme Montagne} - {Axe horizontal} - {Axe vertical}
M - 1 - 1
```

Enfin, le plus important pour les aventuriers, les trésors. Plusieurs trésors peuvent être présents sur une même case; le nombre de trésors sur une même case est indiqué dans le fichier d'entrée de l'exercice par la ligne suivante :

```
# {T comme Trésor} - {Axe horizontal} - {Axe vertical} - {Nb. de trésors}
T - 0 - 3 - 2
```

Exemple pour une carte de 3 x 4 :

```
C - 3 - 4
M - 1 - 1
M - 2 - 2
T - 0 - 3 - 2
T - 1 - 3 - 1
```

Que l'on peut représenter sous la forme suivante :

```
 .     .     .
 .     M     .
 .     .     M
T(2)  T(1)   .
```

Les aventuriers

Un aventurier est caractérisé par sa position sur la carte et son orientation (nord, sud, ...). Il ne peut se déplacer que d'une case à la fois, dans la direction définie par son orientation. Ceci dit, il peut changer d'orientation en pivotant de 90° vers la droite ou la gauche. Il débute son parcours avec une orientation (**N**ord, **S**ud, **E**st, **O**uest), et une séquence de mouvements (**A**vancer, tourner à **G**auche, tourner à **D**roite) prédéfinies. Ils ne sont pas montagnards pour un sou, et ne peuvent donc pas traverser une case montagne.

Exemple de séquence de mouvement :

AGGADADA deviendra : avancer, tourner à gauche, tourner à gauche, avancer, tourner à droite, avancer, tourner à droite, avancer.

Les aventuriers présents sur la carte sont indiqués dans le fichier d'entrée de l'exercice sous la forme suivante :

```
# {A comme Aventurier} - {Nom de l'aventurier} - {Axe horizontal} - {Axe vertical} - {Orientation} - {Séquence de mouvement}
A - Indiana - 1 - 1 - S - AADADA
```

Exemple pour une carte de 3 x 4 :

Au départ :

```
 .     .     .
 .     A     .
 M     .     .
 .     .     .
```

A l'arrivée, étape indiquée entre parenthèses :

```
 .     .     .
 .     .     .
 M    (1)    .
 A    (2)    .
```

On remarquera que l'aventurier reste bloqué en orientation **N**ord, à cause de la montagne. Dans ce cas précis, l'aventurier ignore les mouvements bloquants et poursuit l'exécution de la séquence.

Si l'aventurier passe par dessus une case **T**résor, il ramasse un trésor présent sur la case. Si la case contient 2 trésors, l'aventurier devra quitter la case puis revenir sur celle-ci afin de ramasser le 2ème trésor.

Il ne peut y avoir qu'un aventurier à la fois sur une même case. Les mouvements des aventuriers sont évalués tour par tour. En cas de conflit entre mouvements sur un même tour, c'est l'ordre d'apparition de l'aventurier dans le fichier qui donne la priorité des mouvements.

Ce qu'il faut réaliser

Lire le fichier d'entrée

Le programme doit être capable de lire le fichier d'entrée de l'exercice.

Note : une ligne débutant par un '**#**' est un commentaire et doit être ignorée.

Exemple :

```
C - 3 - 4
M - 1 - 0
M - 2 - 1
T - 0 - 3 - 2
T - 1 - 3 - 3
A - Lara - 1 - 1 - S - AADADAGGA
```

Que l'on peut représenter sous la forme suivante :

```
.           M           .
.           A(Lara)     M
.           .           .
T(2)        T(3)        .
```

Simuler les mouvements des aventuriers

Le programme doit être capable d'exécuter les mouvements des différents aventuriers en respectant les contraintes de l'exercice, de gérer la collecte des trésors et de restituer le résultat final de la simulation.

Dans l'exemple précédent, Lara collecte 3 trésors et finit son parcours en (0 - 3).

Ecrire le fichier de sortie

Le programme doit être capable d'écrire un fichier contenant le résultat final de la simulation.

Note : une ligne débutant par un '**#**' est un commentaire et doit être ignorée.

Voici le format de sortie :

```
C - 3 - 4
M - 1 - 0
M - 2 - 1
# {T comme Trésor} - {Axe horizontal} - {Axe vertical} - {Nb. de trésors restants}
T - 1 - 3 - 2
# {A comme Aventurier} - {Nom de l'aventurier} - {Axe horizontal} - {Axe vertical} - {Orientation} - {Nb. trésors ramassés}
A - Lara - 0 - 3 - S - 3
```

Que l'on peut représenter sous la forme suivante :

```
.           M           .
.           .           M
.           .           .
A(Lara)     T(2)        .
```