

## PROJET TNSI - LA GROTTTE

Dans ce jeu, des pions avancent dans une grotte représentée par une grille à deux dimensions dont chaque case est soit un mur, soit un espace vide, un espace vide pouvant contenir au maximum un pion à un instant donné.

Les pions apparaissent l'un après l'autre à une position de départ, et disparaissent lorsqu'ils atteignent une case de sortie. Chaque pion a une coordonnée verticale et une coordonnée horizontale désignant la case dans laquelle il se trouve, ainsi qu'une direction (gauche ou droite). Les pions se déplacent à tour de rôle, toujours dans l'ordre correspondant à leur introduction dans le jeu, de la manière suivante :

- si la case immédiatement en-dessous est libre, le pion descend d'une ligne ;
- sinon, si la case immédiatement devant est libre (dans la direction du pion concerné), le pion avance d'une case ;
- enfin, si aucune de ces deux conditions n'est vérifiée, le pion se retourne.

Pour programmer l'évolution d'une colonie de pions, on propose une structure avec une classe **Pion** pour les pions, une classe **Case** pour les cases de la grotte et une classe principale **Jeu** pour les données globales.

- La classe principale **Jeu** contient un attribut **grotte** contenant un tableau à deux dimensions de cases, et un attribut **pions** contenant un tableau des pions actuellement en jeu.

Son constructeur initialise la grotte, par exemple à partir d'une carte donnée par un fichier texte (voir l'exemple ci-dessous), où **#** représente un mur, où les pions apparaissent au niveau de la case vide de la première ligne, et **O** représente la sortie.

```
# #####
#                                     #
##### #####
#           #           #
# #####           #
#                                     O
##### #####
#           #
#####
```

Cette classe fournit notamment les méthodes suivantes :

- **affiche(self)** affiche la carte avec les positions et directions de tous les pions en jeu ;
- **tour(self)** fait agir chaque pion une fois et affiche le nouvel état du jeu ;
- **demarre(self)** lance une boucle infinie attendant des commandes de l'utilisateur.

Exemples de commandes : **+** pour ajouter un pion, **q** pour quitter, et **Entrée** pour jouer un tour.

- Une classe **Pion** avec des attributs entiers positifs **l** et **c** indiquant la ligne et la colonne auxquelles se trouve le pion, et un attribut **direction** valant 1 si le pion se dirige vers la droite et  $-1$  si le pion se dirige vers la gauche. Il sera aussi utile d'avoir un attribut **jeu** pointant sur l'instance de la classe **Jeu** pour laquelle le pion a été créé, pour accéder au terrain et à la liste des pions.

Cette classe fournit notamment les méthodes suivantes :

- `__str__(self)` renvoie '>' ou '<' selon la direction du pion ;
- `action(self)` déplace ou retourne le pion ;
- `quitte(self)` retire le pion du jeu.
- La classe **Case** contient un attribut **terrain** contenant le caractère représentant la caractéristique de la case (mur, vide, sortie), et un attribut **occupant** contenant l'éventuel pion présent dans cette case et **None** si la case est libre. Cette classe fournit notamment les méthodes
  - `__str__(self)` renvoie le caractère à afficher pour représenter cette case ou son éventuel occupant ;
  - `libre(self)` renvoie **True** si la case est peut recevoir un pion (elle n'est ni un mur, ni occupée) ;
  - `depart(self)` retire le pion présent ;
  - `arrivee(self, pion)` définit **pion** comme occupant de la case, ou le fait sortir du jeu si la case était une sortie.