TP 7 : manipulations de matrices

PTSI Lycée Eiffel

22 janvier 2015

1 Quelques fonctions sur les tableaux.

On désignera dans ce TP par le mot tableau une variable Python constituée d'une liste de listes (qui ne sont pas toutes nécessairement de la même longueur). Un tel tableau sera une matrice si chacune des listes le constituant a la même longueur.

1. Écrire une fonction Python prenant comme argument un tableau t et déterminant s'il s'agit ou non d'une matrice.
2. Écrire une fonction Python prenant comme argument un tableau t et calculant la longueur de sa plus grande ligne. En déduire une fonction prenant comme argument un tableau t et le transformant en matrice en ajoutant des 0 sur toutes les lignes « incomplètes ».
3. Écrire une fonction Python prenant comme arguments un tableau t et un nombre x et déterminant le nombre de fois où la valeur x apparait dans le tableau t. Modifier cette fonction pour qu'elle affiche non seulement le nombre d'occurrences de x, mais aussi la liste des coordonnées des éléments du tableau égaux à x.
4. Écrire une fonction prenant comme argument un tableau t et déterminant le nombre d'éléments de t (attention aux lignes de longueurs différentes).
5. Écrire une fonction prenant comme argument un tableau t et calculant la moyenne de tous les éléments de t.

2 Application à la programmation d'un démineur.

Si vous ne connaissez pas les règles du jeu du démineur, renseignez-vous ! La grille de démineur sera représentée en Python par une matrice contenant des chiffres compris entre 0 et 9 (les chiffres de 0 à 8 représentant le nombre de mines des cases adjacentes, et le 9 représentant les mines).

1. Écrire une fonction Python qui prend comme arguments trois entiers n, p et k et qui créée une matrice à n lignes et p colonnes avant d'y insérer k nombres 9 à des emplacements aléatoires, le reste étant constitué de zéros. Attention, les mines doivent toutes être sur des cases distinctes. On pourra utiliser certaines fonctions du module random si besoin.
2. Écrire une fonction Python prenant comme argument une grille et deux entiers i et j, et qui compte le nombre de voisins de la case (i, j) (située sur la ligne i et la colonne j) contenant des 9.
3. En déduire une fonction prenant comme argument une matrice remplie de 0 et de 9 (comme celle qu'on a créée à la question 1), et remplaçant les zéros par le nombre de mines adjacentes à la case correspondante.
4. Écrire un programme permettant d'effectuer une partie de démineur (en mode texte, c'est pas très beau, mais c'est plus simple). Le programme procédera de la façon suivante :

   - il demande au joueur des valeurs de n, p et k (nombre de lignes, nombre de colonnes, nombre de mines).

- il créée une matrice représentant la grille de démineur comme vu à la question précédente.
- il créée une deuxième matrice de même taille ne contenant que des 9 (cette deuxième matrice représente la grille de jeu du joueur, ici le 9 correspond aux cases qui n'ont pas encore été explorées).
- il créée une variable contenant le nombre de cases sans mines encore non explorées (initialement, $np - k$).
- à chaque tour de jeu, on affiche la grille du joueur, puis on lui demande de donner les coordonnées d'une case de la grille (un numéro de ligne, un numéro de colonne). Si cette case a déjà été explorée on ne fait rien. Si c'est une case où se trouve une mine, on arrête le jeu. Sinon, on modifie le 9 de la grille du joueur en le remplaçant par le nombre de mines adjacentes à la case qu'on vient d'explorer. On diminue aussi d'une unité le nombre de cases restant à explorer.
- le jeu s'arrête quand il ne reste plus de cases à explorer (ou si on a explosé avant, bien évidemment). On peut afficher un message de félicitations à ce moment.

5. améliorer le programme pour qu'on ait le choix, à chaque tour de jeu, entre explorer une nouvelle case, ou poser un drapeau (marqueur indiquant une position où on pense avoir repéré une mine), qui sera par exemple représenté dans la grille par le caractère M.
6. améliorer le programme pour que, quand on découvre un zéro dans la grille, il démine automatiquement les cases adjacentes (ce que fait le démineur Windows par exemple).
7. Pour les plus forts en Python, programmer une interface graphique à l'aide des modules tkinter ou Pygame.

3 Un autre jeu faisant intervenir des tableaux.

Pour ceux qui n'aiment pas le démineur, voici un autre « jeu » pouvant être modélisé par des matrices. Il s'agit du jeu de la vie de Conway. Une matrice est initialement remplie avec des 0 (case « morte ») et des 1 (case « vivante »). Cette matrice va évoluer au fil du temps de la façon suivante :

- si une case vivante a strictement moins de deux voisines vivantes, elle meurt à l'étape suivante.
- si une case vivante a strictement plus de trois voisines vivantes, elle meurt à l'étape suivante.
- si une case a exactement deux voisines vivantes, elle n'évolue pas au tour suivant.
- si une case morte a exactement trois voisines vivantes, elle devient vivante au tour suivant.

Écrire un programme Python prenant comme arguments une matrice et un entier n et calculant la grille obtenue après n étapes de ce jeu, ainsi que la liste du nombre de cases vivantes après chaque étape. On pourra bien entendu réutiliser certaines fonctions de la partie précédente. Comme dans le cas du démineur, les plus motivés pourront ensuite tenter de créer une interface graphique dans laquelle l'évolution sera visible étape par étape. Accessoirement, vous pouvez aller farfouiller sur le web (chez vous, bien sûr, pas pendant le TP), pour aller voir les drôles de choses qu'on peut faire avec ce jeu.