Tutorat Python

JUNG Frédéric : frédéric.jung@etu.u-bordeaux.fr

Partie 2

16/10/2018 2:19 PM

Exercice 1:

> **Créer une animalerie :**

Pour celà vous devez créer un menu qui vous permettra de demander à l'utilisateur ce qu'il souhaite faire suivant ces options :

- Ajouter un animal
- Afficher les animaux **(On ne souhaite pas afficher une liste mais bien chaque animaux de cette liste)**. Les annimaux doivent être afficher sus la forme suivante :

```
1 --> NomAnimal1
2 --> NomAnimal2
etc...
```

- Quitter le programme (voir `import sys` et `sys.exit()`)

Bonus :

- **Supprimer un animal :** demander à l'utilisateur d'entrer le nom d'un animal qu'il souhaite supprimer. Si cet animal existe bien il est supprimé de l'annimalerie.

Exercice 2

> Reprendre l'exercice 1 avec des fonctions. **Le main ne doit contenir que des appels de fonctions.**

Exercice 3

> Reprendre l'exercice 2. Chaque animal de l'animalerie doit maintenant être associé à un poids entré par l'utilisateur.

La fonction d'affichage affiche maintenant les animaux sous la forme :

```
1 --> NomAnimal1        Poids : PoidsAnimal1
2 --> NomAnimal2        Poids : PoidsAnimal2
etc...
```

Exercice 4

Reprendre l'exercice 3 avec un dictionnaire et utiliser les méthodes des dictionnaires pour :

- afficher les animaux
- afficher la moyenne des poids des animaux

Partie 2 : Grille de jeu

Exercice 1

Utiliser les **listes à deux dimensions** pour générer une grille 10*10 (non graphique) sur laquelle se déplace une joueur. La position initiale du joueur sur la grille est en position (x=5,y=5).

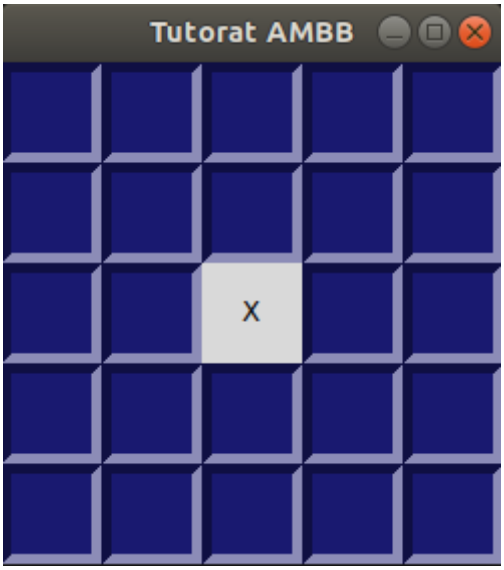
Implémenter une fonction de déplacement pour déplacer le joueur. La nouvelle position du joueur doit être affiché à la fin de chaque déplacement. Le joueur ne doit pas sortir de la grille.

Exercice 2

Afficher graphiquement la grille dans votre terminal.

Bonus

Utiliser Tkinter pour réaliser l'interface graphique. Vous pourvez obtenir un résultat comme celui là :



Partie 3 : lire un fichier

Exercice 1

Lire le fichier iris.csv et calculer et implémenter les deux fonctions suivantes :

- `read.csv(separator,path)` retourne une liste contenant les toutes les lignes du document csv.

```
def read_CSV(separator,path):
    list_lines= []
    # ...
    return list_lines
```

Méthodes utiles : `open()` , `close()`

- `get_sum(col1,col2)` : retourne la somme de la colonne 1 et la somme de la colonne 2

Méthodes utiles : `split()` , `rstrip()`

Iris dataset source : https://archive.ics.uci.edu/ml/datasets/iris

---