Arbres : travaux pratiques

On utilise l'implémentation objet des arbres avec une seule classe présentée dans le cours.

Exercice 1

Ouvrir le fichier arbre.py.

1. Générer un arbre aléatoire avec a = genere_arbre(4) et représenter cet arbre sur papier.

2. Générer un arbre aléatoire avec a = genere_arbre_numerotation(4, p=1) et représenter cet arbre sur papier.

Dans les exercices suivants, on importera le fichier arbre.py avec : from arbre.py import *
Afin d'écrire des codes compatibles avec une implémentation fonctionnelle, on utilisera les fonctions est_vide, etiquette, gauche, droite plutôt que les méthodes correspondantes.
On testera chaque fonction à l'aide du module doctest en exécutant le script.

Exercice 2

Ouvrir le fichier arbre_exercice2.py.

1. Écrire la fonction est_feuille(arbre) renvoyant True si un arbre est une feuille, False sinon.
2. Écrire la fonction compter_feuilles(arbre) renvoyant le nombre de feuilles d'un arbre.
3. Écrire la fonction taille(arbre) renvoyant la taille d'un arbre.
4. Écrire la fonction hauteur(arbre) renvoyant la hauteur d'un arbre.
5. Écrire la fonction ajoute(lst1, lst2) prenant deux listes d'entiers et renvoyant la liste obtenue en ajoutant entre eux les nombres de même positions.
   Ainsi, ajoute([2, 1, 3], [3, 0, 4, 6]) renvoie [5, 1, 7, 6].
6. Écrire la fonction compter_noeud_niveau(arbre) renvoyant le nombre de nœuds pour chaque niveau.

Exercice 3

Ouvrir le fichier arbre_exercice3.py.
Écrire les différents parcours en affichant les étiquettes de chaque nœud séparées d'un espace avec print(etiquette(arbre), end=' ').
On pourra utiliser les files de la bibliothèques collections : **from** collections **import** deque

- file = deque([]) crée une file vide.
- **if** file: convertit file en booléen prenant la valeur False si file est vide, True sinon.
- file.append(e) enfile l'élément e.
- file.popleft() défile un élément et le renvoie.

Ces opérations s'effectuent en temps constant.

Exercice 4

Ouvrir le fichier arbre_exercice4.py.

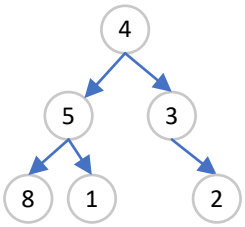
1. Écrire la fonction conversion_iovif convertissant un arbre en implémentation objet en un arbre en implémentation fonctionnelle, comme (1, (2, None, None), (3, None, None)).
2. Écrire les fonctions etiquettef, gauchef, droitf, renvoyant respectivement l'étiquette, le sous-arbre gauche et le sous-arbre droit d'un arbre en implémentation fonctionnelle.

3. Écrire la fonction conversion_ifvio convertissant un arbre en implémentation fonctionnelle en un arbre en implémentation objet.

Exercice 5

Écrire une méthode de la classe Noeud d'entête **def** represente(self, tiret=''): affichant une représentation textuelle d'un arbre, comme dans l'exemple ci-dessous, l'arbre vide étant représenté par le caractère *.

```
4
-5
--8
---*
---*
--1
---*
---*
-3
--*
--2
---*
---*
```



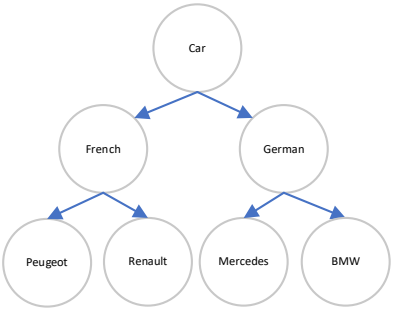
Un affichage sans les arbres vides est plus agréable, mais on ne voit plus si un fils affiché est le fils gauche ou droit, à moins de remplacer le tiret par un symbole différent pour chaque fils, par exemple :

```
4
/5
//8
/\1
\3
\\2
```

Exercice 6

L'application graphviz permet de représenter graphiquement des arbres :
http://www.graphviz.org/download/
On y accède en Python avec la bibliothèque graphviz : https://pypi.org/project/graphviz/
Exemples d'utilisation : https://graphviz.readthedocs.io/en/stable/examples.html
Pour simplifier l'utilisation de graphviz, on va uniquement s'intéresser aux arbres dont tous les nœuds ont des contenus différents. Dans le cas contraire, on doit utiliser un identifiant différent pour chaque nœud et enregistrer le contenu de chaque nœud dans une étiquette (label), ce qui complique grandement la tâche.

1. En s'inspirant de l'exemple hello.py, afficher le graphe suivant.
2. Écrire une fonction montrer_arbre(arbre) qui affiche un arbre en utilisant graphviz.
   Tester cette fonction en générant un arbre avec la fonction genere_arbre_numerotation.



Installation de graphviz sur Windows

J'ai eu un peu de mal à faire fonctionner graphviz sur mon pc. Voilà une procédure qui fonctionne :
http://www.graphviz.org/download/ : j'ai pris le build cmake
Rajouter le chemin vers graphviz/bin dans le path et taper dot -c dans une invite de commandes en mode administrateur.
Puis installer la bibliothèque Python graphviz : https://pypi.org/project/graphviz/