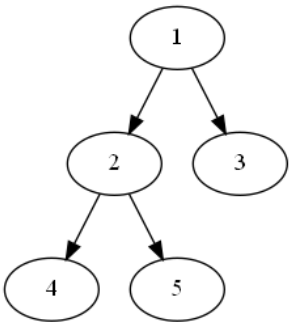
Arbres 2/3 : Avec python (garanti sans ordinateur)

1. Des arbres avec python

Exemple

Voici un exemple de code permettant de construire l'arbre $a1$ ci-contre:

```
a1 = Arbre(1)
b = Arbre(2)
c = Arbre(3)
d = Arbre(4)
a1.ajoute(b, c)
b.ajoute(d, Arbre(5))
```

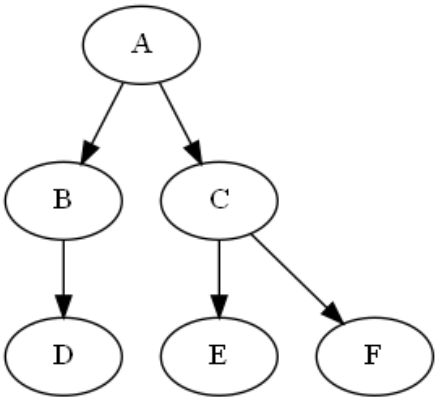


Exercice 1 : Construction d'arbres

1. Dessiner l'arbre $a2$ construit par le code suivant :

```
a2=Arbre(0)
a2.ajoute(Arbre(5),a1)
```

2. Donner le code permettant de dessiner l'arbre ci-contre



3. Dessiner l'arbre $a4$ construit par le code suivant, sachant que `a4.enfants` est la liste des enfants de `a4` .

```
a4 = Arbre(1)
a4.ajoute(Arbre('A'), Arbre('B'), Arbre('C'))
for i in range (3):
    a4.enfants[i].ajoute(Arbre(3*i), Arbre(i+5))
```

Exercice 2 :

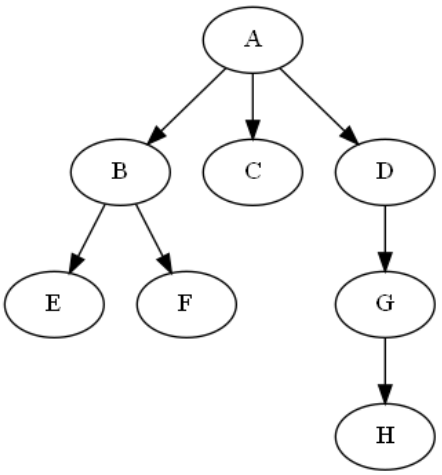
1. Représenter les arbres A,B et D construits par le code ci-dessous:

```
A = Arbre('A')
B = Arbre('B')
C = Arbre('C')
D = Arbre('D')
E = Arbre('E')
F = Arbre('F')
G = Arbre('G')
H = Arbre('H')
B.ajoute(E, F)
D.ajoute(G)
A.ajoute(B, C, H)
G.ajoute(H)
```

2. Représenter l'arbre A après l'instruction `A.enfants[2] = D`

Exercice 3 : On admet que si `A` est un arbre :

- `A.enfants` est la liste des enfants de `A` .
- `A.label` est l'étiquette de `A` .



1. Quels sont le type et la valeur de `A.label` ?

2. Quels sont le type et la valeur de `A.enfants` ?

3. Quels sont le type et la valeur de `A.enfants[0]` ?

4. Quels sont le type et la valeur de `B.enfants[0].label` ?

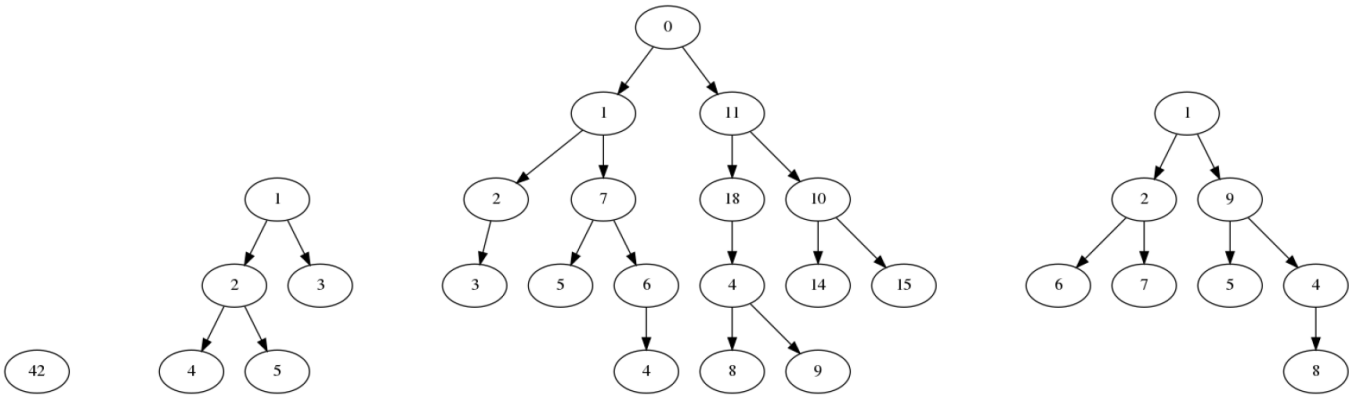
5. Quels sont le type et la valeur de `A.enfants[0].enfants[1].label` ?

2. Récursivité

Rappel : Une fonction est récursive si le code qui la définit contient un appel à elle-même.La structure d'arbre se prête bien à l'utilisation de fonctions récursives.

Exercice 4:

Dans cet exercice, on considère les arbres `a0` , `a1` , `a5` et `a6` respectivement affichés ci-dessous:



1. Compléter le tableau ci-dessous :

	`a0`	`a1`	`a5`	`a6`
Taille				
Hauteur				
Nombre de feuilles				
Plus grande étiquette				

2. Voici une première fonction qui prend un arbre en paramètre :

```
def est_une_feuille(arbre):
    if arbre.enfants==[]:
        return True
    else:
        return False
```

- Est-ce une fonction récursive ?
- Quelles sont les valeurs de `f1` , `f2` , `f3` et `f4` suivantes ?
  - `f1 = est_une_feuille(a0)` :
  - `f2 = est_une_feuille(a1)` :
  - `f3 = est_une_feuille(a1.enfants[0])` :
  - `f4 = est_une_feuille(a1.enfants[1])` :

3. Voici une autre fonction qui prend également un arbre en paramètre :

```
def mystere(arbre):
    if est_une_feuille(arbre):
        total=1
    else:
        total=0
        for sous_arbre in arbres.enfants:
            total=total+mystere(sous_arbre)
    return total
```

- Est-ce une fonction récursive ?
- Donner les valeurs de :
  - `mystere(a0)` :
  - `mystere(a1)` :
  - `mystere(a5)` :
  - `mystere(a6)` :
- Proposer un nom plus adapté à cette fonction :

4. On veut écrire une fonction `somme(arbre)` qui prend en paramètre un arbre et qui renvoie la somme des étiquettes de tous les noeuds de l'arbre.

- Quelles seront les valeurs de `somme(a0)` et `somme(a1)` ? :
- Proposer un code pour cette fonction :

5. Ecrire le code d'une fonction `e_max(arbre)` qui prend en parametre un arbre et qui renvoie la valeur de l'étiquette la plus grande.