INFO-H-100 - Programmation
TP 14 — Recherche de fichiers dupliqués

Lorsque l'exercice demande d'écrire une fonction, écrivez la fonction demandée et testez-la avec plusieurs valeurs pertinentes.

Ex. 1. Ecrire une fonction qui compte le nombre de fichiers et de dossiers immédiatement présents dans le dossier fourni en argument. La fonction renvoie une paire `(nbFichiers, nbDossiers)`.

```
>>> curdir = os.getcwd()
>>> print compteContenu(curdir)
(3, 2)
```

Ex. 2. Ecrire une fonction récursive qui affiche le nom complet (chemin inclus), la taille et la date de modification de tous les fichiers contenus dans un dossier fourni, y compris dans les sous-dossiers de celui-ci.

```
>>> afficheFichiersComplet(curdir)
Tue Jan 31 15:48:21 2012    3952781  D:\test\dico.txt
Wed Nov 24 23:26:32 2010      11776  D:\test\Excel.xls
Wed Nov 24 23:26:32 2010      11776  D:\test\sstest1\Excel.xls
Tue Jan 31 15:48:21 2012    3952781  D:\test\sstest2\sstest3\dico.txt
Wed Nov 24 23:26:32 2010      11776  D:\test\sstest2\sstest3\Excel.xls
Wed Nov 24 23:24:39 2010         20  D:\test\sstest2\WinRAR.rar
Wed Feb 15 18:34:42 2012       3198  D:\test\tp14.py
```

Ex. 3. Ecrire une fonction qui renvoie la taille d'un fichier ou d'un dossier fourni. La taille d'un dossier et définie, récursivement, comme étant la somme des tailles des fichiers et des sous-dossiers qu'il contient.

```
>>> taille('.')
7943715L
```

Ex. 4. Ecrire une fonction `findDup(dir)` qui affiche les fichiers dupliqués apparaissant dans le dossier fourni (ainsi que dans ses sous-dossiers).

```
>>> findDup(curdir)
Excel.xls           11776  Wed Nov 24 23:26:32 2010  D:\test
Excel.xls           11776  Wed Nov 24 23:26:32 2010  D:\test\sstest1
dico.txt          3952781  Tue Jan 31 15:48:21 2012  D:\test
dico.txt          3952781  Tue Jan 31 15:48:21 2012  D:\test\sstest2\sstest3
Excel.xls           11776  Wed Nov 24 23:26:32 2010  D:\test
Excel.xls           11776  Wed Nov 24 23:26:32 2010  D:\test\sstest2\sstest3
```

Deux fichiers seront considérés comme dupliqués s'ils possèdent le même nom, la même taille et le même date de dernière modification (Ils doivent se trouver, bien sûr, dans des dossiers différents). Pour découvrir les fichiers dupliqués, il faut parcourir tous les fichiers du dossier et des sous-dossiers et les accumuler dans un dictionnaire qui permettra, efficacement, de retourner voir si un fichier a déjà été rencontré ou non.

La clé de recherche de ce dictionnaire sera constituée d'un tuple formé du nom, de la taille et de la date du fichier. La valeur associée sera le chemin d'accès de ce même fichier.

Pour chaque fichier rencontré, il s'agira d'aller voir dans le dictionnaire s'il est déjà présent (avec un autre chemin !). S'il n'y est pas, il doit être rajouté. S'il y est déjà, ce fichier sera reconnu comme dupliqué. Il sera affiché avec son chemin propre. Le fichier correspondant dans le dictionnaire sera, lui aussi, affiché.

Bonus : Si le fichier venait à apparaître une troisième fois, lui seul serait alors affiché (par exemple, le fichier `D:\test\Excel.xls` n'est affiché qu'une seule fois comme ci-dessous).

```
>>> findDup(curdir)
Excel.xls           11776  Wed Nov 24 23:26:32 2010  D:\test
Excel.xls           11776  Wed Nov 24 23:26:32 2010  D:\test\sstest1
dico.txt          3952781  Tue Jan 31 15:48:21 2012  D:\test
dico.txt          3952781  Tue Jan 31 15:48:21 2012  D:\test\sstest2\sstest3
Excel.xls           11776  Wed Nov 24 23:26:32 2010  D:\test\sstest2\sstest3
```

INFO-H-100 - Programmation
TP 14 — Recherche de fichiers dupliqués
Corrections

Solution de l'exercice 1:

```
import os

def compteContenu(rep):
  nbF = 0
  nbD = 0
  for f in os.listdir(rep):
    if os.path.isfile(f):
      nbF += 1
    else:
      nbD += 1
  return (nbF, nbD)
```

Solution de l'exercice 2:

```
import os
import time

def afficheFichiersComplet(rep):
  """
  Affichage des fichiers contenus dans le dossier dir et ses sous-dossiers
  Affichage avec formatage de la taille et date de modification
  """
  for f in os.listdir(rep):
    nomComplet = os.path.join(rep, f)
    if os.path.isdir(nomComplet):
      afficheFichiersComplet(nomComplet)
    else:
      taille = os.path.getsize(nomComplet)
      date   = os.path.getmtime(nomComplet)
      s = "%s %10d  %s" % (time.ctime(date), taille, nomComplet)
      print s
```

Solution de l'exercice 3:

```
import os

def taille(f):
  if os.path.isdir(f):
    t = 0
    for sf in os.listdir(f):
      t += taille(os.path.join(f,sf))
    return t
  else:
    return os.path.getsize(f)
```

Solution de l'exercice 4:

```
import os
import time

def findDup(rep):
  fichiers = {}
  aux(fichiers, rep)

def aux(fichiers, rep):
  for f in os.listdir(rep):
    nomComplet = os.path.join(rep, f)
    if os.path.isdir(nomComplet):
      aux(fichiers, nomComplet)
    else:
      checkDup(fichiers, nomComplet)

def checkDup(fichiers, f):
  """
  Verifie si f est deja present dans le dict fichiers.
  Si oui -> c'est un duplique. Il est affiche (avec l'original)
  Sinon  -> il est ajoute dans le dict
  """
  chemin  = os.path.dirname(f)
  cle = (os.path.basename(f), os.path.getsize(f), os.path.getmtime(f))
  if cle in fichiers:
    affiche(cle, fichiers[cle])
    affiche(cle, chemin)
  else:
    fichiers[cle] = chemin

def affiche(cle, chemin):
  (nom, taille, date) = cle
  s = "%-15s %9d  %s  %s" % (nom, taille, time.ctime(date), chemin)
  print s
```

Bonus

```
def checkDup(fichiers, f):
  """
  Verifie si f est deja present dans le dict fichiers.
  Si oui -> c'est un duplique. Il est affiche
  Sinon  -> il est ajoute dans le dict
  L'eventuel double dans fichiers ne s'affiche qu'une fois.
  Ensuite son chemin est mis a None
  """
  nomfich = os.path.basename(f)
  chemin  = os.path.dirname(f)
  cle = (nomfich, os.path.getsize(f), os.path.getmtime(f))
  if cle in fichiers:
    cheminOrig = fichiers[cle]
    if cheminOrig != None:
      affiche(cle, cheminOrig)
    affiche(cle, chemin)
    fichiers[cle] = None
  else:
    fichiers[cle] = chemin
```