LS4: Travaux pratiques 8

7 avril 2014

1 Système de gestion de fichiers

Les exercices suivant porte sur le système de gestion de fichiers (SGF). Pour interagir avec le SGF, vous aurez besoin du module os. Vous pouvez vous référer à la librairie standard de python stdtypes [1].

1.1 Commandes Unix

Question 1

Que fait la commande `os.getcwd()` ?

q1 :

- ☐ affiche le chemin absolu jusqu'au répertoire personnel
- ☐ affiche le chemin relatif jusqu'au répertoire personnel
- ☐ affiche le chemin absolu jusqu'au répertoire courant
- ☐ affiche le chemin relatif jusqu'au répertoire courant

Que fait la commande `os.listdir()` ?

q2 :

- ☐ liste les entrées du répertoire courant
- ☐ liste les entrées du répertoire courant sauf '.' et '..'
- ☐ liste les entrées du répertoire courant sauf les fichiers cachés

Qu'est-ce qui est vrai après l'exécution de la suite d'instructions suivante ?

```
os.mkdir('A')
os.chdir('A')
```

q3 :

- ☐ Le répertoire 'A' est un sous-répertoire du répertoire courant.
- ☐ Le répertoire courant est 'A'
- ☐ Le répertoire 'A' a été créé

Qu'est-ce qui est vrai après l'exécution de la suite d'instructions suivante ?

```
os.mkdir('A')
os.rename('A', 'B')
os.rmdir('B')
```

q4 :

- ☐ Les répertoires 'A' et 'B' sont des sous-répertoires du répertoire courant.
- ☐ Le répertoire 'B' est un sous-répertoire du répertoire courant
- ☐ Le répertoire courant est 'B'
- ☐ Les répertoires 'A' et 'B' ont été créés, puis 'B' a été détruit
- ☐ Le répertoire courant a les mêmes entrées qu'avant l'exécution de la suite d'instructions

1. `http://docs.python.org/3/library/os.html`

Question 2

Exécuter le code python du fichier `arborescence.py` qui crée une arborescence de répertoire racine A. Voir arborescence.py. [2]

La fonction `access(ref, droits)` du module os permet de tester les droits en lecture, écriture et exécution de l'utilisateur sur le fichier de référence ref. Pour tester les permissions, il faut que le paramètre droits soit la disjonction (OR) d'une ou plusieurs des constantes suivantes : R_OK (droit en lecture), W_OK (droit en écriture) et X_OK (droit en exécution).

L'utilisateur a-t-il le droit en lecture sur le répertoire 'A' ?

q5 :

☐ oui

☐ non

L'utilisateur a-t-il le droit en écriture sur le répertoire 'A' ?

q6 :

☐ oui

☐ non

L'utilisateur a-t-il le droit en exécution sur le répertoire 'A' ?

q7 :

☐ oui

☐ non

La fonction `chmod(ref, droits)` du module os permet de modifier les droits du fichier de référence ref.

Essayer de vous déplacer dans le répertoire B. Pourquoi n'y avez-vous pas accès ?

q8 :

☐ l'utilisateur n'a pas les droits en lecture sur 'A'

☐ l'utilisateur n'a pas les droits en écriture sur 'A'

☐ l'utilisateur n'a pas les droits en exécution sur 'A'

☐ l'utilisateur n'a pas les droits en lecture sur 'B'

☐ l'utilisateur n'a pas les droits en écriture sur 'B'

☐ l'utilisateur n'a pas les droits en exécution sur 'B'

Quels sont vos droits sur le répertoire B ?

q9 :

☐ r

☐ w

☐ x

Faites ce qu'il faut pour pouvoir lister le contenu de B. Quel est le nom du répertoire contenu dans B ?

q10 :

Nom du répertoire :

Essayer de créer le répertoire C dans le sous-répertoire de B. Pourquoi est-ce impossible ?

q11 :

☐ l'utilisateur n'a pas les droits en lecture sur le sous répertoire de 'B'

☐ l'utilisateur n'a pas les droits en écriture sur le sous répertoire de 'B'

☐ l'utilisateur n'a pas les droits en exécution sur le sous répertoire de 'B'

1.2 Parcours de l'arborescence des fichiers

La fonction `os.stat(path)` retourne les statuts du fichier dont la référence path est passée en paramètre. Elle retourne un objet dont les champs regroupent les informations sur le fichier. Par exemple le champ `st_ino` correspond au numéro d'inoeud, le champs `st_nlink` au nombre de liens physiques sur le fichier et le champs st_size au nombre d'octets occupés par le fichier. Le fonction `os.path.join(ref, nom)` retourne la concaténation de la référence ref, de '/' et du nom de fichier nom. La fonction `os.path.split(path)` retourne le 2-uplet (ref, nom) où path est la concaténation de ref, '/' et nom.

2. `Fichier fourni avec le sujet.`

Question 1

Ecrire une fonction `taille(ref)` qui retourne la somme des tailles des fichiers du répertoire dont la référence est passée en paramètre. La référence par défaut est '.'

q12 :

Question 2

Exécuter le code suivant dans l'interpréteur python.

```
for path,dirs,files in os.walk("."):
    print(path+" contient les repertoires "+str(dirs))
    print("    et les fichiers "+str(files))
```

Quel type de parcours est effectué ?

q13 :

☐ parcours en largeur

☐ parcours en profondeur

Ecrire une fonction `du_bs(ref)` qui affiche la taille totale d'une arborescence de fichiers dont la réference du répertoire racine est passé en paramètre. La référence par défaut est '.'. On fera l'hypothèse que l'arborescence ne contient pas de lien symbolique, et le minimum de liens physiques. Vous pouvez comparer votre résultat avec la commande UNIX du -s

q14 :

Question 3

Ecrire la fonction `du_liens(ref)` qui fait la même chose que la fonction du_bs en prenant en compte les liens physiques multiples. Si une arborescence contient plusieurs liens physiques sur un même fichier, la taille du fichier ne doit compter qu'une fois. On rappelle que deux liens physiques sur un même fichier ont même numéro d'inoeud.

q15 :

1.3 Recherche de fichiers

Question 1

Ecrire une fonction `list_jpeg(ref)` qui retourne la liste les fichiers JPEG contenus dans une arborescence de fichiers dont la réference du répertoire racine est passée en paramètre. La référence par défaut est '.'.

q16 :

Ecrire une fonction `range_jpeg(dest,src)` qui range dans un répertoire dont la référence est passée par le paramètre `dest` (et le crée au préalable si nécessaire), un lien physique pour chaque image jpeg contenue dans une arborescence de fichiers dont la référence du répertoire racine est passée par le paramètre `src`. La référence par défaut est '.'. Chaque lien portera le même nom que celui donné dans l'arborescence à l'image sur laquelle il pointe, mais on uniformisera la casse en le mettant en minuscule.

q17 :

Question 2

Ecrire une fonction `find(expr, ref)` qui prend en paramètre une expression régulière et une référence de répertoire et affiche les références absolues des fichiers dont le nom est contenu dans l'expression régulière.

q18 :

Ecrire une fonction `find_home_diese` qui affiche tous les fichiers contenus dans l'arborescence de racine le répertoire personnel de l'utilisateur dont le nom contient le symbole #.

q19 :

Ecrire une fonction `find_home_java` qui affiche tous les fichiers contenus dans l'arborescence de racine le répertoire personnel de l'utilisateur dont le nom commence par une suite quelconque de caractères non espacés et terminant par '.java'.

q20 :

Ecrire une fonction `nettoie(ref)` qui prend en paramètre une référence de répertoire et efface de l'arborescence de racine ref tous les répertoires et fichiers vides, ainsi que tous les répertoires ne contenant que des fichiers vides. Il faudra pour cela faire un parcours de l'arborescence des feuilles vers la racine.

q201 :

2 Processus

Les exercices suivant portent sur les processus. Pour gérer les processus en python vous aurez besoin du module subprocess. Vous pouvez vous référer à la librairie standard de python stdtypes [3].

Question 1

Une instance de la classe subprocess.Popen représente un processus. Pour lancer un processus, il faut faire appel au constructeur de cette classe en lui passant en argument une liste contenant la commande et ses options sous forme de chaînes de caractères. Par exemple, l'instruction :

```
p=subprocess.Popen(['ls','-l'])
```

lance un processus qui exécute la commande UNIX 'ls -l'. On peut ensuite envoyer un signal au processus en invoquant la méthode send_signal(num) où num est le numéro du signal que l'on veut envoyer. Par exemple le numéro 2 correspond au signal SIGINT et le numéro 9 au signal SIGKILL.

La calculatrice dc permet de faire des calculs en ordre polonais inversé, en prenant les commandes sur l'entrée standard. Par exemple, si on tape :

```
dc 2 2+
```

alors la calculatrice affiche 4 lorsqu'on tape la lettre p (print).

Ecrire une fonction `mon_dc` qui lance un nouveau processus qui exécute dc, s'endort pendant 20 secondes, puis tue le processus. Lancer la fonction, faire des calculs et interroger le shell avec la commande ps pour connaître les processus en cours.

q21 :

Question 2

On peut aussi rédiriger l'entrée standard de dc sur un fichier de calcul. La commande UNIX est alors :

```
dc < calcul
```

On peut également rediriger la sortie et la sortie d'erreur vers des fichiers. La commande UNIX est alors :

```
dc < calcul > calcul.out 2> calcul.err
```

Le constructeur de la classe subprocess.Popen peut prendre des options qui permettent de rediriger l'entrée et les sorties de la commande sur des fichiers. L'instruction :

```
p=subprocess.Popen(cmd, stdin=entree, stdout=out, stderr=err)
```

3. `http://docs.python.org/3/library/subprocess.html`

où entree, out et err sont des descripteurs de fichiers, redirige l'entrée sur le fichier de descripteur entree, la sortie sur le fichier de descripteur out et la sortie erreur sur le fichier de descripteur err.

Créez un fichier calcul qui contient les lignes suivantes :

```
40
260+
10/
p
12*
p
7
0/
```

Ecrire une fonction `dc_fics(calcul)` qui prend en paramètre un fichier de calcul, lance dc avec en entrée ce fichier et en sortie un fichier portant le même nom avec l'extension '.out' et en sortie erreur un fichier portant le même nom avec l'extension '.err'.

q22 :

Question 3

Si l'on veut communiquer par tubes, il faut alors positionner les options stdin, stdout et stderr du constructeur subprocess.Popen, à subprocess.PIPE. Il faut par ailleurs positionner l'option universal_newlines du constructeur à True pour que la lecture et l'écriture dans les tubes se fassent en mode texte et non en mode binaire. Pour communiquer avec le processus lancé, vous pouvez appeler la fonction communicate sur l'objet représentant le processus. communicate prend en paramètre une chaîne de caractères correspondant à l'entrée de la commande et retourne deux chaînes de caractères correspondant au résultat de la commande exécutée sur l'entrée et à la sortie erreur de cette commande.

Ecrire la fonction `dc_tubes` qui lance dc en communiquant par tubes et affiche les sorties des tubes stdout et stderr.

q23 :

Question 4

Ecrire la fonction `tube(cmd1, entree, cmd2)` qui exécute l'instruction UNIX :

```
cmd1 < entree | cmd2
```

où entree est une chaîne de caractères.

q24 :