Projet bash

A. OBJECTIFS DU TP

Mettre en œuvre les connaissances acquises dans l'ensemble du module (cours et TP) en développant une série de scripts en `bash`.

B. LA COMMANDE PS

1. Examiner la commande ps -edf. Que fait-elle ?
2. Analyser chaque champ du résultat de la commande *ps -edf*. Nous nous intéresserons plus particulièrement aux colonnes PID, PPID et CMD.
3. Qu'est-ce que le PID ? Le PPID ? Quelle est la relation entre le PID et le PPID ? Observer le dossier **/proc** et regardez le nom des dossiers.

Le but du projet est de créer l'arborescence résultant de la commande *ps -edf*. Tout comme chaque répertoire a un répertoire parent, chaque processus en a un aussi. Il faudra donc créer l'arborescence sur ce modèle. Les répertoires auront le nom des PID correspondant. Dans chaque répertoire, un fichier texte ***process*** sera créer contenant le nom du processus (CMD)

C. ARBO_PS.SH

Nous appellerons cette commande arbo_ps.sh et nous l'implémenterons d'une option d'aide -h ou --help qui renverra l'aide suivante :

```
NOM
      arbo_ps – Produire une arborescence correspondant au résultat de la
commande ps -edf.

SYNOPSIS
      arbo_ps.sh [OPTION]... [ -d REPERTOIRE ]

DESCRIPTION
       arbo_ps.sh crée une arborescence correspondant au résultat de la
commande ps –edf. Les répertoires sont hiérarchisés suivant leur PID et leur
PPID. Chaque répertoire contient un fichier contenant le nom du processus
auquel le PID correspond.
```

```
Les options disponibles sont :

-d <chemin>
    Si l'option est absente, la commande crée l'arborescence à partir
    du répertoire courant.

-v, --verbose
    Afficher les différentes étapes de la commande
    (Exemple : « Phase 1 : Traitement des arguments »

 -n, --pid <entier>
    Créer l'arborescence depuis le PID indiqué et non depuis 0.
```

Voici un exemple pour mieux comprendre. Ci-joint, un extrait d'une commande ps -edf :

…
cli 5403 5399 0 13:18 ? 00:00:00 sshd: cli@pts/4
cli 5406 5403 1 13:18 pts/4 00:00:00 -bash
cli 5457 5406 1 13:18 pts/4 00:00:00 ps -edf
…

La commande suivante est exécutée :

arbo_ps.sh -n 5403

Nous obtiendrons donc une arborescence de cette forme (résultat d'un ls -lR) :

5403:
total 4
-rw-r--r-- 1 cli sit 0 3 janv. 2017 5403.txt
drwxr-xr-x 3 cli sit 4096 3 janv. 2017 5406

5403/5406:
total 4
-rw-r--r-- 1 cli sit 0 3 janv. 2017 5406.txt
drwxr-xr-x 2 cli sit 4096 3 janv. 2017 5457

5403/5406/5457:
total 0
-rw-r--r-- 1 cli sit 0 3 janv. 2017 5457.txt

D. NOTATION & CONSEILS

Vous ne devez utiliser QUE des commandes et des notions vues en cours de R104. Pas de commande *awk* par exemple.

Une attention particulière sera portée aux commentaires.

La gestion des erreurs doit être présente. Au moins 5 erreurs distinctes doivent être traitées.

Pour tous commentaires supplémentaires destinés au correcteur, veuillez joindre au projet un fichier texte décrivant les problèmes rencontrés. De même si le projet n'est pas achevé, veuillez préciser l'état d'avancement de celui-ci, ainsi que les difficultés ayant conduit à l'échec du projet, et évaluer le temps nécessaire qu'il aurait fallu pour le terminer.

Conseils :

Vous pouvez utiliser des fichiers temporaires. Le répertoire /tmp peut être un excellent chemin pour cela. N'oubliez pas de les effacer.

Pour la gestion des arguments, inspirez-vous de la dernière partie du cours « Annexe »

Combinez les commandes *head* et *tail* pour récupérer une ligne précise d'un fichier.

Les commandes suivantes sont correctes et doivent donc fonctionner :
./arbo_ps.sh -n 0 -d /tmp -v

Aide à la programmation :

Pour lire un fichier ligne par ligne, et les traiter (ici les lire) :

```
while read line
do
        echo $line
done < fich.txt
```

ou :

```
IFS='\n'
For line in $(cat fich.txt)
```