ISS - Initiation aux Systèmes d'exploitation et au Shell
LU2IN020
Partiel du 10 décembre 2020

Nom : **Prénom :** **Numéro de groupe :**

Aucun document autorisé pendant l'épreuve
Les téléphones portables, les montres connectées et autres appareils doivent être rangés dans votre sac.
Le barème n'est donné qu'à titre indicatif, pour vous permettre de juger de la difficulté des questions.
Attention : l'énoncé est imprimé recto-verso sur **10 pages**.

Hypothèse pour l'ensemble de l'examen : Pour simplifier, si les questions n'indiquent pas le contraire, on supposera que tous les exécutables sont bien présents dans le répertoire de l'exercice et que les droits nécessaires à leurs exécutions sont attribués à l'ensemble des utilisateurs.

Exercice 1 : Questions de cours (7,5 points)

Question 1 – 1,5 point

Dessinez, tel que présenté au début de chaque cours, l'ensemble de la mémoire d'un processus en indiquant ses différentes zones, les registres spéciaux, ainsi que l'ensemble des données de l'OS permettant de gérer le fonctionnement de ce processus.

Question 2 – 1,5 point

Dessinez sous forme d'un automate le cycle de vie d'un processus depuis sa création jusqu'à sa mort. Votre schéma devra indiquer l'ensemble des états possibles, ainsi qu'une raison pour chaque changement d'état. Votre schéma devra aussi différencier les systèmes fonctionnant en mode *temps partagé*, de ceux fonctionnant en mode *batch*. On ne considérera pas ici l'état *zombie*.

Question 3 – 0,5 point

Pour chacune des fonctionnalités suivantes, indiquez si elle est matérielle ou logicielle (vous n'avez pas ici à justifier votre réponse) :

1. la pile d'exécution des processus
2. le mode U et le mode S
3. les droits sur les fichiers enregistrés sur le disque

Question 4 – 1 point

Qu'est-ce qu'une commande *builtin* ? Illustrer votre propos à l'aide de chronogrammes en comparant ces commandes à des commandes classiques.

Question 5 – 0,5 point

Pourquoi la commande `cd` ne peut-être qu'une commande *builtin* ?

Question 6 – 0,5 point

Listez les trois fonctionnalités majeures d'un système d'exploitation présentées en cours.

Question 7 – 1 point

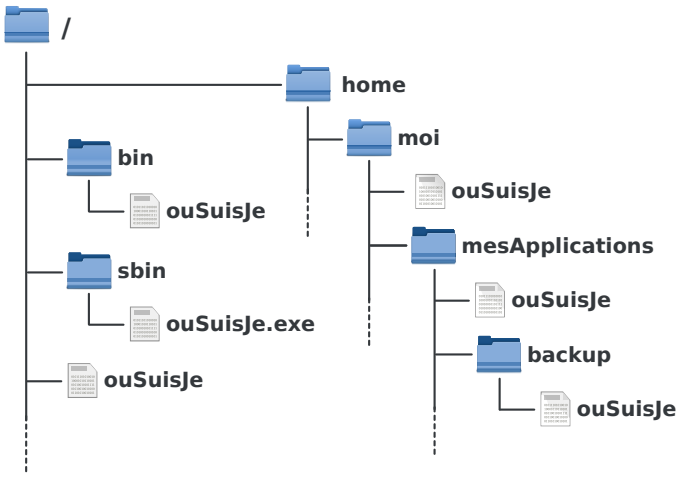
Donnez dans l'ordre tous les ancêtres d'un processus exécutant un bash dans un terminal graphique. Parmi eux, vous indiquerez ceux qui appartiennent à l'utilisateur `root`.

Question 8 – 1 point

À la PPTI vous souhaitez laisser l'accès au fichier "solution.txt" se trouvant dans votre *home* à un camarade de promo pour qu'il puisse le copier. Donnez la ou les commandes qui lui donnent les droits nécessaires tout en s'assurant qu'il ne pourra pas lister le contenu de votre *home*.

Exercice 2 : La route est longue, mais le chemin est libre (6 points)

Dans cet exercice, on considère l'exécutable `ouSuisJe` qui, une fois lancé, affiche le chemin complet du répertoire qui le contient. On suppose aussi que l'arborescence du système (représentée ci-dessous) contient plusieurs instances de cet exécutable.



Question 1 – 0,5 point

Donnez, en le justifiant, l'affichage produit par l'appel à l'exécutable `ouSuisJe` qui suit la séquence de commandes suivante :

```
moi@pc /home/moi $ echo $PATH
/sbin:/bin:/usr/bin:/usr/share/bin
moi@pc /home/moi $ pwd
/home/moi
moi@pc /home/moi $ ouSuisJe
```

Question 2 – 0,5 point

On désire maintenant être certain que l'instance exécutée soit celle qui se trouve dans le répertoire `mesApplications`. Modifiez la dernière commande (et uniquement la dernière commande) en conséquence.

Question 3 – 0,5 point

Quelle commande, exécutée au préalable, permettrait d'arriver au même résultat sans avoir à modifier la séquence initiale ? Vous veillerez dans votre réponse à garder les bonnes pratiques vues en TP.

Question 4 – 0,5 point

Comment peut-on appliquer cette modification à tous les terminaux qui lanceront un **`bash`**.

Question 5 – 2 points

Une dernière alternative pourrait être de supprimer toutes les copies de l'exécutable susceptibles d'être choisies par le *Bash*. On souhaite donc implémenter un script `supprimerDesChemins.sh` qui efface toutes les instances d'un programme (dont le nom sera passé en paramètre) qui pourrait être choisi par le *Bash*.

Donnez une première version qui pour tous les répertoires considérés :

1. vérifiera la présence de l'exécutable ;
2. le supprimera, le cas échéant.

Pour simplifier, on supposera que le script est toujours appelé avec un paramètre et que vous avez toujours les droits de supprimer les fichiers.

Question 6 – 2 points

Pour finir, on veut implémenter une nouvelle version du script `supprimerDesChemins.sh` qui évitera de tester tous les répertoires pour ne parcourir que les répertoires qui contiennent l'exécutable ciblé.

Donnez une solution qui utilise la commande `myWhereis`. Cette commande maison ressemble à la commande `type -a` ou à la commande originale `whereis` mais est plus simple à parser. Ainsi, elle affiche à l'écran toutes les instances d'un binaire susceptibles d'être choisies par le bash de la façon suivante :

```
moi@pc /home/moi $ myWhereis uneCommande
/rep1/rep2/uneCommande /rep3/uneCommande /uneCommande
```

Vous veillerez ici à traiter le cas où le programme passé en paramètre n'est pas accessible du tout.

Exercice 3 : Si loin, si proche (4 points)

Question 1 – 1 point

Que fait chacune des commandes suivantes :

```
sed -E '/iss/!d' unFichier.txt
sed -E -n '/iss/p' unFichier.txt
```

Question 2 – 1 point

Que fait chacune des commandes suivantes :

```
sed -E '/[^:]*:[^:]*/d' unFichier.txt
sed -E '/^[^:]*:[^:]*$/d' unFichier.txt
```

Question 3 – 1 point

Que fait chacune des commandes suivantes :

```
sed -E -n 's/s( *)/x\1/g' unFichier.txt
sed -E -n 's/s( +)/x\1/g' unFichier.txt
```

Question 4 – 1 point

Que fait chacune des commandes suivantes :

```
sed -E 's/a/b/g;s/b/a/g' unFichier.txt
cat unFichier.txt | tr ab ba
```

Exercice 4 : C'est mon mien (4 points)

Dans ce dernier exercice, on s'intéresse au parsing de la commande `ls -l`, dont un exemple est donné ci-dessous. Vous êtes ici libres d'utiliser toutes les commandes de votre choix. Lors de la correction, une attention particulière sera accordée à la simplicité de la réponse.

```
moi@pc /home/moi $ ls -l dir/
total 12
-r--rw-r-- 1 moi monGroupe 10 10 déc. 01:53 ma_photo
drwx------ 2 moi monGroupe 40 10 déc. 03:52 subdir/
-rw------- 1 toi tonGroupe 12 10 déc. 04:23 texte
drwxr----- 2 moi monGroupe 40 10 déc. 01:31 un_autre_subdir/
-rw-r----- 1 moi monGroupe 10 10 déc. 00:11 une_liste
```

Question 1 – 2 points

Pour commencer, implémentez un script `combien.sh` qui affiche sur le terminal le nombre de fichiers d'un répertoire (dont le nom est passé en paramètre) qui appartiennent au lanceur du script et pour lesquels il a les droits en écriture. Vous ferez donc attention à la première ligne et au répertoire (préfixé par `d` dans l'affichage du `ls`), ainsi qu'à la présence d'un paramètre.

Au besoin, vous pourrez vous aider de la commande `whoami` qui affiche le nom de l'utilisateur.

Question 2 – 2 points

On veut maintenant compter ce type de fichier pour un ensemble de répertoires. Implémentez un script qui :

— lit une chaîne de caractères sur le clavier contenant une liste de noms de répertoire séparés par un espace (on ne considère ici que des noms sans espaces).

— affiche le nombre total de tous les fichiers de ce type contenus dans l'ensemble des répertoires de la liste (lue sur le clavier)

Attention, tous les mots de la chaîne lue sur le clavier qui ne correspondraient pas à un répertoire devront être ignorés.