Janvier 2016

Système 1ère année

Pas de calculatrice

Tous documents autorisés

Exercice 1 (Nommage et caractères de substitution : 2 + 1 + 2 = 5 pts)

1. Je veux que le répertoire */users/paviot/Documents/tmp* devienne le répertoire courant. Quelle commande dois-je taper ? Ai-je utilisé un nom absolu ou un nom relatif dans cette commande ? Ce répertoire est considéré comme étant le répertoire courant pour le reste de l'exercice.
2. Donnez la commande permettant de copier tous les fichiers du répertoire avec l'extension *.txt* dans le répertoire père (le père du répertoire courant, bien sûr).
3. Expliquez ce que fait la commande suivante, en précisant si les noms utilisés sont des noms absolus ou relatifs : « *mv /tmp/rapport-V??.* .* ».

Exercice 2 (Droits : 1 + 2 + 1 = 4 pts)

1. Je veux lire un fichier *exemple.txt* dont les droits sont *rwxr-x--x*. Dans quel(s) cas puis-je le faire ?
2. Nous voulons modifier les droits du fichier *liste.txt*. Donnez la commande (une seule commande !) permettant à la fois :
   - d'ajouter le droit en écriture pour les membres du groupe,
   - de retirer le droit en exécution pour les autres.

   Vous utiliserez la forme symbolique. Quelle est la condition pour que vous puissiez réaliser cette opération ?
3. Dans la question précédente, on demande d'utiliser la forme symbolique. Etait-il envisageable d'utiliser la notation octale ? Si oui, réécrivez la commande en utilisant cette notation, sinon, pourquoi est-ce impossible ?

Exercice 3 (Variables : 1 + 2 = 3 pts)

1. Créez une variable *TEST* qui prend la valeur : *ceci est un essai.*
2. D'habitude la variable *PATH* ne contient que des noms absolus. Est-il possible de mettre des noms relatifs ? Avec quelle conséquence ?

Exercice 4 (Commandes find et grep 1 + 1 = 2 pts)

Que font les commandes suivantes ?

1. grep -r "^#!/bin/bash" /*
2. find / -newer fic.txt –print

Questions (1 + 2 + 2 + 1 = 6 pts)

1. Supposons que la commande « *cp *.txt ..* » donne le message d'erreur suivant : *cp : omission du répertoire « liste.txt/ »*. Donnez sa signification.
2. Expliquez ce qu'il se passe quand un script utilise un paramètre, qu'il référence par *$2* par exemple, et qu'aucun paramètre n'a été donné dans la commande lançant son exécution. Est-ce une erreur ?
3. Expliquez ce que font les commandes suivantes :
   - echo $10
   - echo ${10}
4. Ecrivez une commande (et une seule !) permettant de trier un fichier *file*, puis de l'afficher page par page.

sort file | more

less

UNIVERSITÉ PARIS DESCARTES

IUT

DÉPARTEMENT INFORMATIQUE

Écrire très lisiblement (en capitales)

NOM : JAREY

Prénom : Grégoire

DISCIPLINE : SYS

Date de l'épreuve : 17/01/2017

Année : 1e Groupe : 101

Très bien

NOTE DE 0 À 20	APPRÉCIATIONS
16/20	

Ne rien écrire dans cette marge

Exercice 1 (4,5/5)

1) cd /users/pariot/Documents/tmp

2 c'est un nom absolu car il démarre par un /.

0,75 2) cp "*.tst" ..

3) mv permet de déplacer ou renommer des fichiers.
ici il déplace les fichiers se nommant rapport-V??.* se situant dans /tmp (nom absolu). 0,25
(chaque point d'interrogation remplace 1 caractère ↓ remplace une chaîne de caractère)

vers le répertoire courant (.) qui est donc un nom relatif. 0,5

Exercice 2 (4/4)

1) pour lire ce fichier il faut que je soit soit propriétaire (user), soit membre du groupe. Les autres (others n'ayant que le droit en éxecution. 1

2) chmod g+w,o-x liste.txt 1

il faut nécessairement que je soit propriétaire du fichier pour pouvoir effectuer cette modification. 1

3) non, ce n'était pas possible. La forme octale nécessite de savoir quels sont les droits actuels du fichier avant de les modifier (similaire au = en forme symbolique). 1

Exercice 3 (2,5/3)

1) TEST='ceci est un essai' 1

2) la variable PATH contient les chemins absolus vers les commandes externes. De plus, c'est une variable d'environnement, elle se passe donc du père au(x) fils. Mettre des noms relatifs marcherait pour le répertoire père, mais plus pour ses fils. Les fils ne trouveraient donc plus le chemin vers les commandes externes 1,5

Exercice 4 (2/2)

1) sélectionne les lignes commençant (^) par #!/bin/bash dans toute l'arborescence depuis la racine (/*) (-R permet de parcourir récursivement)

2) cette commande affiche (-print) les fichiers situés à la racine (find /) dont la date de création est plus récente (-newer) que celle de fic.txt.

Questions (3/6)

1) la commande cp utilisé sans -R ne copie pas de manière récursive. Si liste.txt/ est un répertoire et qu'il n'est pas vide, il n'est alors pas copié.

2) Lorsqu'un script utilise un paramètre qui ne lui est pas donné à son execution il s'execute en utilisant plusieurs fois le paramètre qui lui a été donné. Cela peut être voulu, ce n'est donc pas nécessairement une erreur.

3) echo $10 → affiche le contenu de la variable 10

echo ${10} → affiche le contenu des variables 1 à 10.

4) sort file | more