
TP N°7 : Les scripts shell

Objectifs	:	<ol style="list-style-type: none">1. Lancer un script shell.2. Ecrire un script avec paramètres3. Utiliser les différentes structures de contrôle du shell
Volume horaire	:	9 heures (3 séances)

I. Partie théorique

Répondre aux questions suivantes.

1. Qu'est-ce qu'un script ?
2. Comment peut-on exécuter un script ?
3. Comment peut-on référencer les paramètres d'un script ?

II. Manipulations pratiques

Exercice 1 :

- Ecrire un script shell qui reçoit trois paramètres et qui affiche leurs valeurs. Un message d'erreur sera affiché si le nombre de paramètres est différent de trois.
- Donner le programme shell qui permet d'afficher le contenu des fichiers donnés comme paramètre. Faites en sorte qu'il demande à chaque fichier une confirmation.
- Ecrire un programme shell qui permet de tester si deux fichiers donnés en paramètres sont liés (commande `test fichier1 -ef fichier2`)
- Ecrire un programme shell qui permet de tester si le fichier1 est plus récent que le fichier2 (les deux fichiers sont donnés en paramètres). (commande `test fichier1 -nt fichier2`)
- Ecrire un programme shell qui permet de tester si le fichier2 est plus récent que le fichier1 (les deux fichiers sont donnés en paramètres). (commande `test fichier1 -ot fichier2`)

Exercice 2 :

Ecrire un programme shell qui reçoit plusieurs paramètres (des noms de fichiers). Pour chaque fichier reçu, il s'agit d'effectuer les tâches suivantes

- Si le fichier est ordinaire vous lui ajoutez le droit de lecture, s'il ne l'a pas déjà.
- Si le fichier est un répertoire vous lui ajoutez le droit « x ».
- Sinon vous réinitialisez toute la chaîne de permission relative au propriétaire à « rwx ».

Exercice 3 :

Ecrire un script qui reçoit comme paramètre un nom de fichier s'il ne reçoit aucun paramètre, il doit afficher un message d'erreur: paramètre manquant. Le fichier reçu comme paramètre sera affiché en indiquant son type et ses droits d'accès.

Exercice 4 :

Le script boucle affiche des nombres à l'écran il sait traiter jusqu'à trois paramètres. Le premier est la valeur de départ, le second est la valeur finale et le 3^{ième} est l'incrément. Si deux paramètres seulement sont spécifiés, l'incrément par défaut sera 1. En présence d'un paramètre unique, la valeur finale sera 20. En absence de tout paramètres on affiche tous les chiffres entre 1 et 20 avec un pas égale à 1.

Exercice 5 :

Ecrivez un programme shell qui permet de renommer une série de fichiers. Lisez le nom du fichier, s'il existe et non vide, vous lisez le nouveau nom, renommez le fichier et bouclez sur une nouvelle lecture, sinon vous affichez un message d'erreur et quittez le programme.

Exercice 6 :

Donnez le programme shell qui permet d'afficher le contenu des fichiers données comme paramètre. Faites en sorte qu'il demande à chaque fichier une confirmation.

Exercice 7 :

Ecrire un programme shell qui reçoit plusieurs paramètres (des noms de fichiers). Pour chaque fichier reçu, il s'agit d'effectuer les tâches suivantes

- Si le fichier est ordinaire vous lui ajouter le droit de lecture, s'il ne l'a pas déjà.
- Si le fichier est un répertoire vous lui ajoutez le droit « x ».
- Sinon vous réinitialisez toute la chaîne de permission relative au propriétaire à «rwx».

Exercice 8 :

Ecrire un script qui reçoit comme paramètre un nom de fichier s'il ne reçoit aucun paramètre, il doit afficher un message d'erreur: paramètre manquant. Le fichier reçu comme paramètre sera afficher en indiquant son type et ses droits d'accès.

Exercice 9 :

Ecrire un script boucle qui affiche des nombres à l'écran il sait traiter jusqu'à trois paramètres. Le premier est la valeur de départ, le second est la valeur finale et le 3^{ième} est l'incrément. Si deux paramètres seulement sont spécifiés, l'incrément par défaut sera 1. En présence d'un paramètre unique, la valeur finale sera 20. En absence de tout paramètres on affiche tout les chiffres entre 1 et 20 avec un pas égale à 1.

Exercice 10 :

- Ecrire un script qui permet d'afficher tous les fichiers qui se trouvent dans le répertoire courant et qui sont vides. (Affich1)
- Changer le script pour qu'il affiche tous les fichiers vides qui se trouvent dans la sous arborescence à partir d'un répertoire cité en argument (effectuer un test sur l'argument pour que le script retourne un message d'erreur si le premier argument n est pas un répertoire). (le script sera appelé Affich2)
- Ecrire un script permettant de supprimer tous les fichiers vides qui se trouvent dans la sous arborescence à partir d'un répertoire cité en argument après avoir affiché leur liste et demandé une confirmation de la suppression. (RamasseMiette)

Exercice 11 :

Ecrire un script Outil qui permet de lancer le programme utilitaire relatif au fichier donné en paramètre. Exemple gcc pour un programme C, vi pour un fichier texte, sh pour un fichier script.

Exercice 12 :

Ecrire un script qui permet d'afficher le squelette d'un menu de la façon suivante :

Menu

- 1 choix 1
- 2 choix 2
- 3 choix 3
- 4 choix 4

0 Quitter

Entrez votre choix : -

Exercice 13 :

- Ecrire un script qui permet de créer un processus qui affiche le message « Vous êtes à l'ISET de Radès » suivi d'un beep sonore toutes les minutes. (Bienvenue)
- Lancer ce script. Que ce passe-t-il ?
- Lancer ce script en arrière plan.
- Repérer le numéro du processus.
- Essayez d'interrompre l'exécution de ce script (mais lancé par votre voisin). Que faut-il changer?
- Arrêter le processus correspondant à l'exécution de Bienvenue.

Exercice 14 :

Ecrire un programme shell pour tester si un fichier est un fichier ordinaire, un répertoire ou spécial.

Exercice 15 :

- Ecrire un script permettant de lancer une suppression par lot tous les fichiers cités en argument seront supprimés après une demande de confirmation.
- Etendre le script pour qu'il fasse les suppressions des fichiers et des répertoires et refuse la suppression des fichiers spéciaux. Une confirmation de la part de l'utilisateur est attendue. Pour le cas des répertoires non vides, Un message sera géré par vous indiquant l'impossibilité de suppression du répertoire non vide.

Exercice 16 :

Ecrire un programme shell qui reçoit comme paramètre deux fichiers. Ce programme permet de tester si les deux fichiers sont liés (par la commande ln) ou non

Exercice 17 :

Créer la commande shell crefic obéissant à la syntaxe suivante : \$ crefic nom quantité

Son rôle est de créer un ensemble de fichiers appelés nom1, nom2, ..., nomN. La création de chaque fichier doit être validée en interactif par l'utilisateur.

Exercice 18 :

Ecrire un script callable soit avec trois paramètres telle que appelé sans paramètre le programme réalisera la lecture au clavier de trois chaînes de caractères ; disposant alors dans tous les cas de trois chaînes, la programme indiquera si les trois chaînes sont identiques, Si deux de ces chaînes sont identiques ou Si elles sont toutes différentes. Rédigez un compte rendu décrivant ce que vous avez fait.

Exercice 19 :

Ecrire un script qui permet d'afficher tous les utilisateurs par groupe d'appartenance.

Group1

utilisateur1

utilisateur 2

Group2

utilisateur3

utilisateur 4

Group3

utilisateur 5

utilisateur 6

Exercice 20 :

Ecrire un script qui permet de copier la sous arborescente d'un répertoire dans un autre. Ce script admet deux paramètres, le premier est la sous arborescence à copier et le deuxième est le répertoire où on va faire la copie.

Rédigez un compte rendu décrivant ce que vous avez fait.