INF203 - Travaux pratiques, séance 2

Scripts shells : redirections, variables, expansion

[INF203] Lors de la dernière séance de TP vous avez utilisé un fichier de commande nommé `installeTP.sh`. Relisez-le et utilisez-le pour créer un répertoire ***TP2*** et y copier les fichiers nécessaires à cette séance.

1 Redirection des entrées et/ou des sorties d'un programme

Exécutez la séquence de commandes suivante :

```
date
date > une_date
cat une_date
```

Exécutez maintenant la commande :

```
ls > une_date
```

[**a**] Que s'est-il passé pour le fichier `une_date` ? ■

Le programme `max2.sh` lit deux entiers au clavier et affiche leur maximum. Lisez-le : pour comprendre certaines lignes, vous aurez besoin du prochain cours, mais vous devriez être capable de deviner leur signification.
[**b**] Notez la taille du fichier `max2.sh`. Exécutez le programme `max2.sh` en notant soigneusement sur votre compte rendu ce qui s'affiche à l'écran, en distinguant à l'aide de deux couleurs ce qui correspond aux sorties du programme, et ce qui correspond à l'écho de ce qui a été saisi au clavier. ■

Créez (avec l'éditeur de fichier de MobaXTerm) un fichier `deux_entiers` contenant uniquement deux entiers, chacun sur une ligne distincte. Essayez la commande suivante pour exécuter `max2.sh` en redirigeant les entrées depuis le fichier `deux_entiers` :

```
./max2.sh < deux_entiers
```

[**c**] Notez les messages d'erreur que vous obtenez si vous exécutez : `./deux_entiers > ./max2.sh`
Donnez-vous les droits en exécution sur le fichier `deux_entiers` et ré-essayez la commande précédente. Quel est alors le message d'erreur obtenu ? Quelle est désormais la taille du fichier `max2.sh` et que contient-il ? Pourquoi ?
■
[**d**] Quelle commande utilisez-vous pour retrouver le fichier `max2.sh` originel ? (Vous en avez besoin pour la suite, si vous ne savez pas faire, demandez de l'aide à votre enseignant). ■

Essayons maintenant de rediriger les sorties de `max2.sh` vers un fichier `resultat` :

```
./max2.sh > resultat
```

[**e**] Pourquoi ne se passe-t-il "rien" ? ■
Quand vous aurez trouvé la réponse à cette question, et réagi en conséquence, vérifiez que le fichier `resultat` contient bien le résultat attendu.
[**f**] Quelle commande tapez-vous pour exécuter **max2.sh** en lisant les entrées dans le fichier `deux_entiers` et en écrivant le résultat dans `resultat1` ? Que se passe-t-il si vous tapez plutôt la commande ci-dessous ?
`./max2.sh > resultat2 < deux_entiers`
Y-a-t-il une différence entre les deux fichiers de résultat ? ■

2 Enchaînement de l'exécution de deux programmes.

Le programme `2entiers.sh` génère aléatoirement deux entiers et les affiche. Exécutez-le deux ou trois fois.

Utilisez ce programme pour fabriquer un fichier `deux_entiers_bis`, en redirigeant ses sorties. Vérifiez que `deux_entiers_bis` contient bien deux entiers. Utilisez `max2.sh` pour afficher le maximum de ces deux entiers en redirigeant ses entrées.
Comme vous l'avez vu en cours et en TD, on peut enchaîner l'exécution de deux programmes ou commandes avec un "pipe" :

```
./2entiers.sh | ./max2.sh
```

Exécutez cette commande quelques fois.

3 Les deux commandes de la semaine : `expr` et `cut`

`expr` : opérations arithmétiques sur deux entiers
Lisez et comprenez le programme `somme.sh`.
[g] Que signifie la syntaxe `$(...)` ? Que fait la commande `expr` ? ■
Sur le même modèle, créez des programmes `difference.sh`, `produit.sh`, `quotient.sh` et `reste.sh` (la commande `cp` est utile pour éviter de tout ressaisir) qui calculent respectivement la différence etc de deux entiers. Les opérateurs pour le quotient et le reste sont respectivement / et %. Testez.

`cut` : découper `ls -l` en rondelles .
Exécutez la commande `ls -l`
Nous allons utiliser la commande `cut` pour afficher la 2ème puis la 5ème colonne des lignes affichées par `ls -l`.

- 2ème colonne. Le séparateur de colonne est l'espace. La commande à utiliser est `cut -d ' ' -f 2`. Essayez : enchaînez (avec —) les exécutions de `ls -l` et de ce `cut`, et vérifiez que vous obtenez le résultat attendu.
- 5ème colonne. Essayez : si vous remplacez 2 par 5 dans la commande ci-dessus, vous n'obtenez pas le résultat espéré. En effet, les colonnes sont parfois séparées par plusieurs espaces. Pour supprimer les espaces exédentaires, il faut utiliser la commande `tr -s ' '`. Enchaînez les exécutions de `ls -l` et de ce `tr` et constatez que les colonnes sont maintenant toutes séparées par un espace unique. Enchaînez maintenant les 3 commandes et vérifiez que vous obtenez bien la cinquième colonne.

4 Instant Suivant

En vous aidant des commentaires explicatifs, complétez le programme `instant_suivant.sh` : ce programme lit un instant au clavier exprimé sous la forme HH:MM:SS (par exemple 09 :30 :25) et calcule l'instant à la seconde qui suit (dans notre exemple 09 :30 :26) et l'affiche à l'écran.
[h] Combien de tests (et quels tests) faut-il faire pour vérifier tous les cas de figure ? ■

[i] Joignez le texte de `instant_suivant.sh` à votre compte rendu. ■

La commande `date` utilisée avec l'option `+%T` permet d'afficher l'heure au format HH:MM:SS. On veut maintenant calculer l'instant suivant de celui donné par la commande `date`. Exécutez les commandes :

```
date +%T
date +%T > une_date
./instant_suivant.sh < une_date
```

[j] Comment pouvez-vous obtenir le même résultat sans créer le fichier intermédiaire `une_date` ? ■

Exercice complémentaire :

Code de retour des commandes

Lorsqu'il y a un problème lors de l'exécution d'une commande unix, cela provoque l'affichage d'un message d'erreur. Exécutez successivement les commandes suivantes, et notez les messages qui sont affichés, et à quel moment :

```
mkdir DIR
mkdir DIR
```

```
rmdir DIR
cd DIR
mkdir DIR
cd DIR
cp ../max2 cop
cp ../../max2 cop
cp cop
```

Outre l'affichage ou non d'un message, les commandes "renvoient" un code entier, qui est différent selon qu'il y a eu ou non un problème lors de l'exécution. On peut connaître ce code en affichant la variable `$?` juste après l'exécution de la commande. Supprimez tous les fichiers du répertoire `DIR` et le répertoire `DIR` lui-même. Refaites la suite de commandes ci-dessus, et après chacune d'elles, exécutez la commande

```
echo $?
```

[k] A quoi correspondent les codes de retour obtenus (erreur/pas erreur) ? ■

Installons le TP

Appellons NOM_A et NOM_B les deux membres de votre binôme. Lors du premier TP, vous avez travaillé sur le compte de NOM_A. Avant de commencer, vous allez recopier le fichier `installeTP.sh` dans l'espace de travail de NOM_B.

[l] Pourquoi NOM_A n'a-t-il pas le droit d'effectuer cette copie ? ■

Connectez-vous sur le compte de NOM_B, créez un répertoire `INF203`, et recopiez-y le fichier `installeTP.sh` du compte de NOM_A.

[m] Quelle commande avez-vous saisi pour effectuer cette copie ? ■

[n] Quelle commande pourriez-vous saisir pour que NOM_A ait le droit d'effectuer cette copie ? Et pour que NOM_B ne puisse plus la faire ? ■