Shell

Exercice 1 :

Écrire des commandes avec **ls** permettant de lister dans un répertoire toutes les entrées dont le nom :

1. commence par la lettre ***t***;
2. comporte au moins un ***d***;
3. commence par ***a*** ou ***t***;
4. comporte un ***a*** puis un ***b***, dans cet ordre mais pas nécessairement contigus.

Exercice 2 :

L'utilisateur Toto tape les commandes suivantes dans un shell :

```
{toto} 1 > ls -l
total 10
drwxr-xr-x 2 toto toto 512 Feb 5 18:22 TP1
drwxr-xr-x 2 toto toto 512 Feb 5 18:22 TP2
drwxr-xr-x 2 toto toto 512 Feb 5 18:22 TP3
drwxr-xr-x 2 toto toto 512 Feb 5 18:22 TP4
-rw-r--r-- 1 toto toto 51  Feb 5 18:22 mystere.sh
{toto} 2 > more mystere.sh
#!/bin/sh
mkdir TP$2
mv TP$1/* TP$2
rmdir TP$1
```

— Q1. Indiquez quelle(s) commandes Toto doit utiliser pour se donner les droits en exécution sur le fichier de commande `mystere.sh`
— Q2. Décrivez en quelques lignes l'effet de la commande `mystere.sh 3 5`
— Q3. Indiquez le contenu du répertoire courant après exécution de cette commande.

Exercice 3 :

Modifiez le fichier de commande copiedir.sh pour qu'il permette :
1. de créer un répertoire TP1 dans votre répertoire principal ;
2. de copier dans ce répertoire l'ensemble des fichiers qui se trouvent dans le répertoire TP1 de votre binôme, dont le nom de login est binome.

Exercice 4 :

Dans un fichier de commande, la commande « if [e1 -gt e2] then ... » permet de tester si l'entier e1 est strictement supérieur à l'entier e2.
En utilisant cette commande, écrivez un fichier de commande permettant d'afficher à l'écran le maximum de deux entiers passés en argument.
Même question pour afficher le maximum de trois entiers passés en argument.

Exercice 5 :

Ecrire un fichier de commandes qui crée deux répertoires Executables et Autres dans le répertoire courant, examine tous les fichiers du répertoire courant et déplace les exécutables dans le répertoire Executables, les autres dans Autres.

Exercice 6 :

Ecrire un script shell qui calcule et affiche la somme des N premiers entiers naturels où N est une valeur

donnée en argument de la ligne de commande. Si aucun argument n'est donné, le script devra afficher un message d'erreur et se terminer.

Exercice 7 :

- Q1. Ecrire un script en shell nommé execute tout.sh qui exécute tout les fichiers ordinaires (pas les répertoires) du répertoire courant ayant les droits en exécution.
- Q2. Comment donnez vous les droits en exécution à votre script ?
- Q3. Si votre script est exécuté dans le répertoire dans lequel il se trouve, comme il a les droits en exécution il va s'exécuter lui-même. Cette exécution va a nouveau entrainer une exécution du script lui-même et ainsi de suite (indéfiniment). Comment faire pour éviter cela ?

Exercice 8 :

Soit le contenu du fichier mon prog3.sh :

```
#!/bin/sh
cd $2
echo $0
mkdir ../$1
cp * $1
```

Que se passe-t-il à l'exécution de la commande :

```
{chapoh} mon_prog3.sh INF122 TP1
```

Exercice 9 :

Ecrire un programme, en langage shell, qui accepte trois arguments (deux entiers m et n, et un nom de fichier nom) et qui exécute deux fois le fichier nom : en redirigeant les entrées depuis le fichier donnéesm et les sorties vers le fichier rrrm, en redirigeant les entréees depuis le fichier donnéesn et les sorties vers le fichier rrrn. Citer quelques messages d'erreurs qu'il est possible d'obtenir lors de l'exécution de ce fichier shell.

Exercice 10 :

Ecrire un programme script (en langage shell) choix.sh, qui accepte 3 arguments, et qui affiche soit le deuxième argument soit le troisième argument, selon que le premier argument est égal à 2 ou à 3.
Que se passe-t-il à l'exécution des commandes suivantes ?

```
choix.sh 3 bonjour bonsoir
choix.sh bonjour bonsoir
choix.sh 2 bonjour
choix.sh 3 bonsoir
```

Exercice 11 :

La ligne #!/bin/bash en tête d'un fichier signifie :

- que le fichier est corrompu
- que le fichier est à interpréter par le programme bash
- que le fichier a été enregistré par le programme bash
- que le fichier est un fichier binaire compilé par bash

Exercice 12 :

Ecrivez un programme shell compile.sh qui nécessite un argument aaa et compile sous ce nom le programme écrit dans le fichier aaa.c, s'il existe. Si le nombre d'arguments est incorrect ou si le fichier aaa.c n'existe pas, un message d'erreur est affiché. On pourra utiliser les opérateurs de test suivants :
xx -eq yy (resp [xx -ne yy]) est vrai si et seulement si les entiers xx et yy sont égaux (resp. différents).
-f nom est vrai si et seulement si un fichier nom est présent dans le répertoire courant.

Exercice 13 :

1) Ecrire un fichier de commande de nom "rangeTP1.sh" qui déplace tous les fichiers terminés par ".c" du répertoire TP1 dans le répertoire TP1/Sources (à créer) et tous les autres dans le répertoires TP1/Divers

(à créer également) et affiche le contenu de ces deux répertoires

2) modifiez "rangeTP1.sh" en un fichier de commandes "rangedir.sh" qui prenne en paramétre le nom du répertoire que l'on veut "ranger" (ex : "rangedir.sh TP2").

modifiez maintenant "rangedir.sh" pour que le suffixe des répertoires à ranger dans "Sources" ne soit pas tjrs ".c" mais soit également un paramétre (ex : "rangedir.sh TP2 sh")

Exercice 14 :

- écrire un fichier de commande qui préfixe par "rep_" tous les répertoires présents dans le répertoire courant.
- écrire un fichier de commande qui efface tous les fichiers exécutables de tous les répertoires du répertoire courant
- écrire un fichier de commande qui calcule et affiche la somme des N premiers entiers, où N est un argument de la ligne de commandes. (ca n'a bien sur pas vraiment d'intérêt en shell ...)

Exercice 15 :

Ecrire un fichier de commande qui prend 3 arguments représentant une heure (heure minute seconde), calcule et affiche l'heure à la seconde suivante.