TP n° 1

Unix - In4001

I) Ordonnancement

Exercice 1 :
Donner l'ordonnancement sur un mono-processeur du système de tâche ci-dessous suivant l'algorithme d'ordonnancement à priorité fixe (Fixed-Priority) :

'	Temps d'arrivée	Durée	Priorité
T1	0	5	2
T2	0	1	1
T3	0	3	4

(les priorités élevées => tâches les plus prioritaires)

Exercice 2 :
Donner l'ordonnancement sur un mono-processeur du système de tâche ci-dessous suivant l'algorithme d'ordonnancement FCFS (First-Come, First-Served) aussi appelé FIFO (First In First Out) :

	Temps d'arrivée	Durée
T1	4	5
T2	2	1
T3	1	3

Refaire l'ordonnancement avec un temps d'arrivée pour la tâche T1 de 1.

Exercice 3 :
Donner l'ordonnancement sur un mono-processeur du système de tâche ci-dessous suivant l'algorithme d'ordonnancement du tourniquet (Round Robin ou RR) :

	Temps d'arrivée	Durée
T1	0	6
T2	0	5
T3	0	10

(quantum de temps fixé à 4)

II) Droits d'accès

Exercice 4 :
A quels droits correspondent les entiers 751, 521, 214 et 150 ?

Exercice 5 :
Que fait la commande "chmod 750 fichier" ? Donner ensuite une autre écriture possible pour cette commande.

Exercice 6 :
Par quels entiers sont codés les droits "rw-r–r–" et "rwxr-xr-x" ?

III) Quelques commandes

Exercice 7 :
Affichez, à l'aide d'une commande, le nombre total de fichiers (fichiers cachés non compris) contenus dans votre répertoire "home". On reprendra l'exemple du cours avec les commandes *ls* et *wc* ;

Exercice 8 :
Modifiez l'apparence de votre prompt lorsque vous avez la main. Et du prompt lorsqu'une commande commencée n'est pas terminée. Fermez le terminal et ré-ouvrez en un, les modifications ont elles persitées ? Essayer de trouver le nom du fichier à modifier pour rendre ces modification persistantes ;

Exercice 9 :
Afficher les alias actuels. Puis définissez les alias suivants : ll (affiche tous les fichiers au format long, fichiers cachés compris), llm (même affichage que ll mais page par page) et rm (demande une confirmation).

IV) Premiers scripts

Exercice 10 :
Afficher, à l'aide d'un script, le chemin de votre répertoire "home". N'oubliez de faire attention à ce que le fichier soit exécutable ;-).

Exercice 11 :
Ecrire un script permettant d'afficher :
- le nombre total d'argument passé au script ;
- la liste des arguments

Rajoutez ensuite des commandes permettant d'afficher les arguments passés au script à l'envers (on se limitera à trois arguments dans un premier temps). Puis modifiez le script pour que cet affichage soit valide quelque soit le nombre d'argument passé au script.

Exercice 12 :
Ecrire un script contenant les trois variables suivantes : PREFIXE, NOM_FICHIER et NOUV_FICHIER. La valeur de la variable NOUV_FICHIER doit être la concaténation de PREFIXE et NOM_FICHIER séparé par "_". Exemple : si PREFIXE vaut "TP1" et que NOM_FICHIER vaut "script.sh", l'affichage de la variable NOM_FICHIER doit être "TP1_script.sh".

Exercice 13 :
Modifiez le script afin qu'il créé un fichier avec le nom NOUV_FICHIER. Le fichier ayant une "extension" .sh il doit être possible de l'exécuter (même s'il ne contient rien dans notre cas). Le script doit indiquer sur le terminal qu'un fichier vient d'être créé. Exemple : "Creation du fichier TP1_script.sh...".

Exercice 14 :
Modifiez le script afin que les variables PREFIXE et NOM_FICHIER soit fixées par l'utilisateur.

Exercice 15 :
Ecrire un script permettant de créer un répertoire avec le nom PREFIXE et de créer un nombre N de fichier à l'intérieur. Le premier argument correspondra à PREFIXE, le deuxième à NOM_FICHIER et le troisième au nombre N de fichier NOUV_FICHIER à créer.

Ces fichiers devront être numéroté (en partant de 0) et auront une "extension" ".sh" (NOM_FICHIER ne devrait donc pas contenir "l'extension"). La numérotation devra apparaître avant "l'extension".

Le script ne devra pas proquer "d'erreur" d'affichage. Par exemple si le répertoire PREFIXE existe déjà, il ne faudra pas qu'il cherche à le créer.

Comment faire pour effacer le répertoire PREFIXE et l'ensemble de son contenu ?

Exercice 16 :
Déplacer vous dans le répertoire PREFIXE. Lancez le script dans ce répertoire (sans l'avoir copier dedans bien sûr...). Trouvez un moyen de pouvoir le lancer sans que l'utilisateur ait besoin de préciser son chemin d'accès. Testez ensuite la solution proposée.

V) Utilisation des fonctions

Exercice 17 :
Ecrire les fonctions *somme*, *soustraction*, *mult* et *div* qui prennent deux nombres en paramètre et affiche le résultat de l'opération correspondante. Toutes variables utilisées par les fonctions doivent être locales. La déclaration des fonctions devra se faire sous le terminal. Afficher ensuite le contenu d'une fonction de votre choix sous le terminal. Effacez ensuite ces fonctions.

Exercice 18 :
Créer un script permettant d'appeler les fonctions précédentes. Tous les appels seront effectués dans le script.

Exercice 19 :
Modifiez le script précédent afin que l'utilisateur puisse appeler l'une des fonctions du script (paramétré par un argument). Les deux paramètres des fonctions (correspondant aux nombres à sommer, multiplier, etc) seront également donnés en argument au script. Exemple d'appel du script : "./fcts_op.sh somme 3 4". Le script devra indiquer à l'utilisateur si la fonction demandée n'est pas définie.