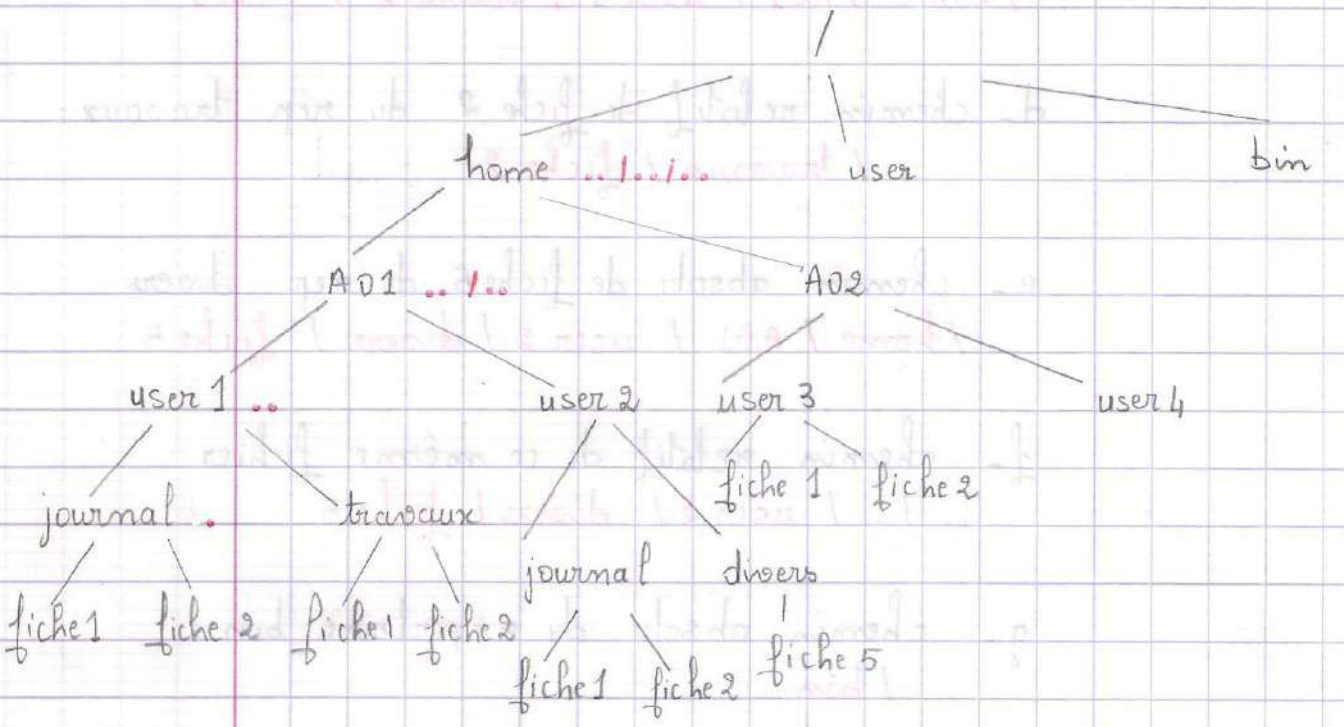
Système d'exploitation

18_09_2014

TD



chemin absolue
=> c'est toujours par rapport à la racine
chemin relatif
=> par rapport au repertoire courant
.. répertoire parent
. répertoire courant

L'utilisateur se trouve actuellement dans le sous-répertoire journal du répertoire user 1

I/_ Donner les chemins suivants :

a_ chemin absolu du répertoire courant :

/ home / AD1 / user 1 / journal

b_ chemin relatif de user 1 :

..

c- chemin absolu de fiche 1 du rép travaux :
/home / A01 / user 1 / travaux / fiche 1

d- chemin relatif de fiche 2 du rép travaux :
.. / travaux / fiche 2

e- chemin absolu de fiche 5 du rép divers
/home / A01 / user 2 / divers / fiche 5

f- chemin relatif de ce même fichier
.. / .. / user 2 / divers / fiche 5

g- chemin absolu du répertoire bin :
/ bin

h- chemin relatif de ce même répertoire
.. / .. / .. / bin

i- chemin absolue de user 3
/ home / A02 / user 3 /

j- chemin relatif de user 3
.. / .. / .. / A02 / user 3

II / - Ecrire les commandes permettant les actions suivantes (sans changer le rép de travail)

a- lister le contenu du répertoire courant:

ls (ou bien ls .)

b- lister le contenu du rép travaux :

ls .. / travaux

c- afficher le chemin absolue du rep courant

pwd

d- créer un répertoire lundi :

mKdir lundi (ou bien mKdir . / lundi)

mieux ↑

attention : mKdir / lundi : le rép va être crée dans la racine

e- créer un fichier vide cr.txt

touch cr.txt

f- copier fiche 1 dans travaux en le nommant fiche 3

cp fiche 1 .. / travaux / fiche 3

source destination

g- copier fiche 5 dans le répertoire courant:

cp .. / .. / user 2 / divers / fiche 5 .

source

Ps: le chemin peut être absolue ou relatif

h/ renomer ce fichier en fichier_user2

mv fiche5 fiche_user2

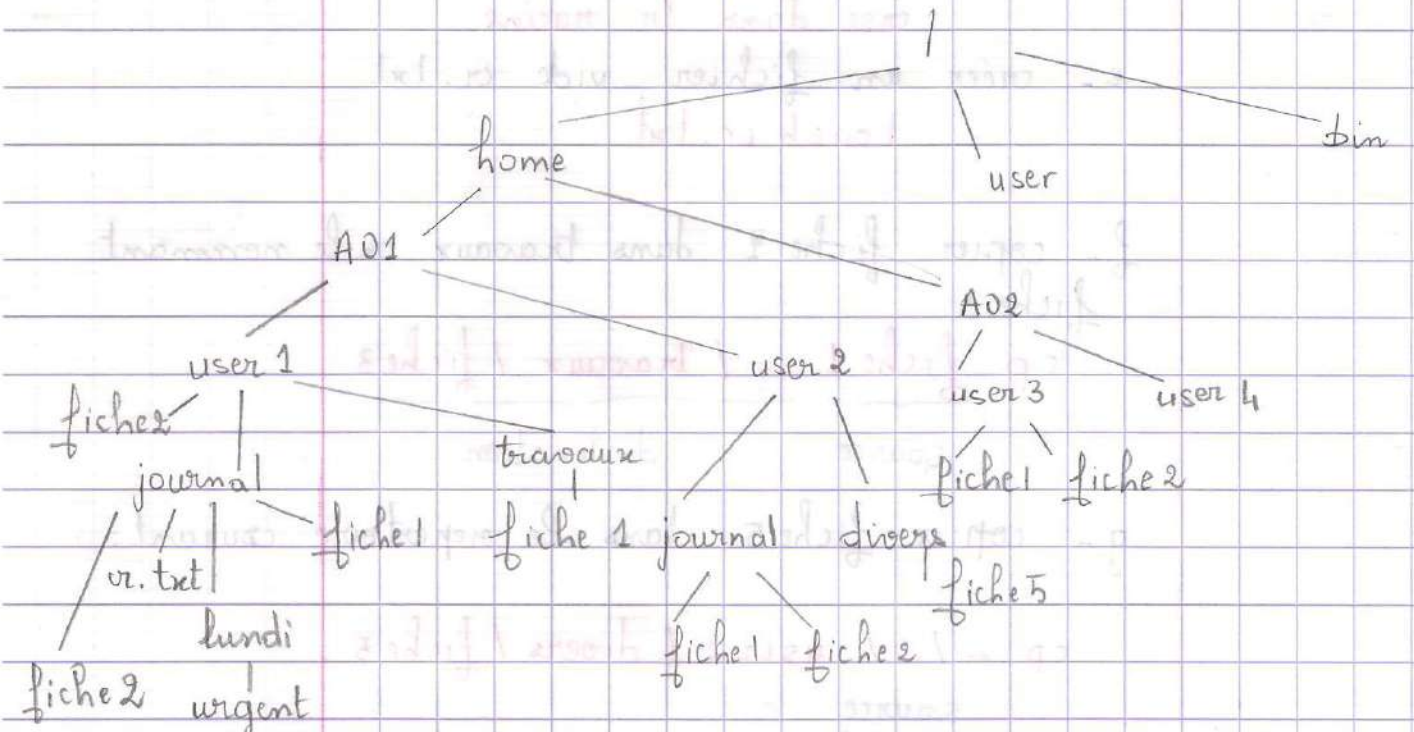
i/ déplacer fiche2 du répertoire courant dans le répertoire user1

mv fiche2

j/ déplacer fiche2 du rép travaux dans le rép lundi en le renommant urgent

mv .. / travaux / fiche2 lundi / urgent

La nouvelle arborescence :



2 - 10 - 2014

exercice 3 : utilisation des droits d'accès

- riri et fifi appartiennent au même groupe (1)
- loulou ∈ groupe 2 (autres)
- "/tmp" contient "correction_projet" accessible en lecture par tous les utilisateurs
- Le répertoire "partage" contient un fichier "projet"
- Le répertoire de travail est le rép "partage"
  (répertoire courant)

configuration 1 :

a/ cp /tmp/correction_projet projet

(droit d'accès de lecture ; droit d'accès de "modification" écriture)

riri - fifi - ~~loulou~~

b/ cp /tmp/correction_projet projetbis

on doit avoir les droits d'accès d'écriture sur "partage"

- ~~fifi~~ - ~~loulou~~

configuration 1 (2 & 3)

c/ chmod 666 projet

(riri)

seul le propriétaire qui peut executer la commande "chmod"

d/ rm projet

riri - fifi - loulou

configuration 2

a/ cp /tmp/ correction_projet projet

riri - fifi - loulou

b/ cp /tmp/ correction_projet projetbis

riri - fifi - loulou

c/

d/ rm projet

riri - fifi - loulou

configuration 3

a/ cp /tmp/ correction_projet projet

→ aucun

b/ cp /tmp/ correction_projet projetbis

riri - fifi - loulou

c/

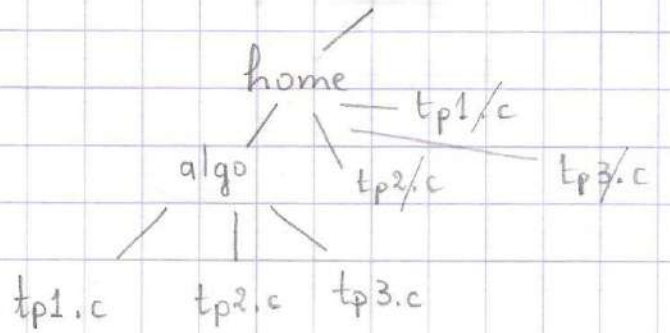
d/ rm projet
riri - fifi - loulou

droit d'execution sur les répertoires seulement si on veut modifier de répertoire de travail.

drw-rw-rw- _____ partage
$ cd partage
affiche un message d'erreur :
"vous n'avez pas de droit d'execution sur le répertoire partage."

exercice supplémentaire :

1- Que font les commandes suivantes, si "algo" est un sous répertoire du répertoire courant :
1°/ mv tp1.c algo
2°/ mv tp2.c algo
3°/ mv tp3.c algo

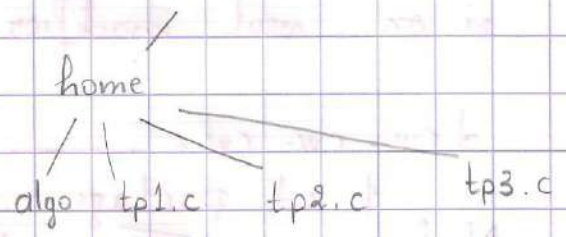


2- si "algo" est un fichier texte du répertoire courant.

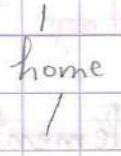
Quel est l'effet de ces trois mêmes commandes?

1°/ mv tp1.c algo renomer tp1.c en algo
2°/ mv tp2.c algo
3°/ mv tp3.c algo

avant



après



algo ← qui contient le "contenu" de tp3.c

16 - 10 - 2014

exercice 2 (La commande chmod)

rw- r-- r-- toto

1/ nouveaux droits d'accès

rwx r-x --- toto

texte symbolique

	utilisateurs
user	u (propriétaire)
group	g (groupe de prop)
other	o (autre)
all	a (tout le monde)

actions
+ (ajout d'un droit)
- (suppression ...)
= (initialisation ...)

droit
r (lecture)
w (ecriture)
x (exécuter)

exemple :

chmod go-x fichier

=> suppression du droit d'execution pour un groupe ou pour autre

chmod u+r-w fichier

=> ajout du droit de lecture

et

suppression du droit d'ecriture

chmod a-x fichier

=> suppression du droit d'execution (x) pour tout le monde (a)

u	g	o
r w [-] r w [x]	1 - [-] r - [x]	r - - [-] - -
+ x	+ x	- r

toto

$ chmod ug + x toto
$ chmod o - r toto

2/ chmod mod1, mod2 ... toto
a/ $ chmod ug + x, o - r toto
b/ $ chmod u = rwx, g = rx, o = toto

3/ écriture numérique (octale)

	u	g	o
symbole	r o x	r - x	- - -
binaire	1 1 1	1 0 1	0 0 0
octal	7	5	0

r → 4 = 2^2
w → 2 = 2^1
x → 1 = 2^0

$ chmod 750 toto

exercice supplementaire

1- Liens physiques :

Lorsque vous modifier les données dans un fichier qui possède des liens physiques :

a/ juste les données de ce fichier sont affectées,

b/ juste les données de ce fichier et les données de ses liens physiques du même repertoire sont affectées.

c/ Les données de ce fichier et les données de ses liens physiques sont affectées puisqu'ils ont des i-noeuds différent

d/ Les données de ce fichier et les données de liens physiques puisqu'ils partagent le même i-noeud

Rq : Lien symbolique aura ses propres i-noeuds

2/ Les commande chgrp , chown :

un utilisateur utilise la commande chgrp pour donner la propriété d'un fichier à un deuxième utilisateur.

Que doit faire l'utilisateur initial pour reprendre le proprieté du fichier ?

justification concernant a et b => ce n'est plus le propriétaire du fichier

a/ l'utilisateur initial exécute à nouveau la commande "chgrp" en précisant que c'est lui le nouv propriétaire

b/ le nouveau proprietaire exécute la commande "chgrp" et precise que l'utilisateur initial est le nouveau propriétaire

c/ L'utilisateur initial exécute la commande "chown" en précisant que c'est lui le nouveau propriétaire

3. La commande "umask"

après l'execution de la commande umask 731
Quels seront les droits d'accès sur les fichiers manuellement créer?

a/ rw- rw- rw-

b/ rwx rw- r--

c/ --- r-- rw-

d/ --- -wx --x

Rq: ls -ld Rep

↳ affiche les informations sur le répertoire "Rep"

4. La commande "chmod"

un fichier "fichier1" a le droits suivants:

r ---- x -w-

la commande chmod 143 fichier1 aura le même effet que:

a. chmod u+r-x, g+r-x, o+w fichier1

b. chmod u=w, g=rw, o=rx fichier1

c. chmod u-r-w, g+r-w, o+r-x fichier1

d. chmod a=x, g=r, o=wx fichier1

e. chmod u+w, g+r-w, o+r-x fichier1

f. chmod u=rw, g=r, o=r fichier1

30-10-2014

exercice 4 : (Programmation Shell)

Question (1, 2, 3, 5)

$ cmd (Le système cherche cette commande dans les repertoires définis dans la variable d'environement PATH)

$ ls-l aucune commande avec le nom ls-l

(PATH doit contenir le chemin vers le répertoire courant)

1/ ./usr/bin
- export/home/
- /bin

-> PATH = $PATH : /usr/bin : /export/home/ : /bin
on utilise les deux points pour séparer les chemins

attention!
PATH = /user/bin : /export/home/ : /bin
=> cette commande va écraser le contenu de la variable PATH.

variable = valeur ← pour affecter on utilise pas $ sauf si on veut affecter le contenu d'une autre variable

pour accéder à la valeur d'une variable on utilise $
echo $ variable

ex : x = 123
variable = $x
echo $ variable => 123

$ which ls
=> /bin/ls) le chemin vers le répertoire qui contient cette commande

21
utilité de & : exécution d'une commande en arrière plan.
comment remettre cette commande en premier plan ?
=> $ fg

exemple 1 :
$ cmd & <= execution de cmd en arr plan
$ fg <= remet cmd en prem plan

exemple 2 :
cmd1 & ; cmd2 & ; cmd3 &
[1] ____ cmd 1
[2] ____ cmd 2
[3] ____ cmd 3
↑
numéro de la tâche

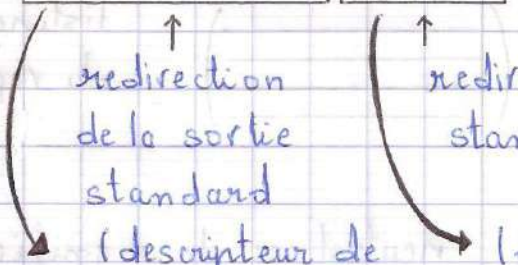
$ fg 2 <- numéro de la tâche à mettre en premier plan

ctrl + Z
$ bg
} pour la remettre en arrière plan.

. cmd1 & ; cmd2 & ; cmd3
=> cmd1 et cmd2 sont exécutées en arrière plan et cmd3 est exécuté en premier plan.
. cmd1 & ; cmd2 & ; cmd3 &
=> les trois commandes sont exécutées en arrière plan

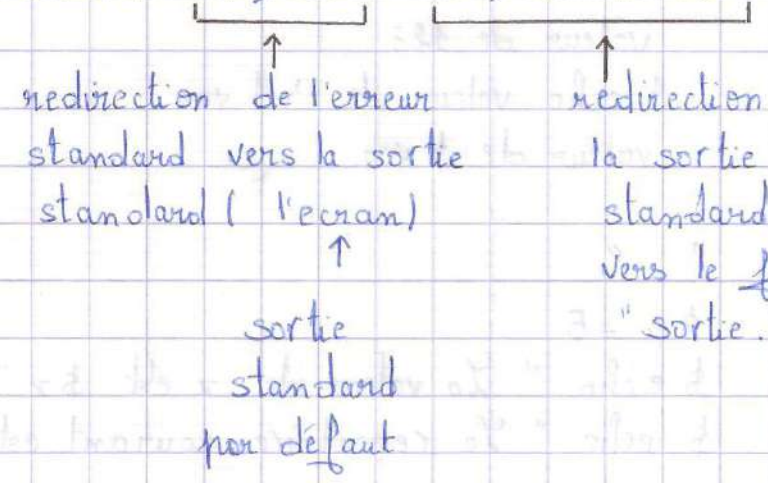
3_ Quelle est la diff entre:
. cmd 1 1> sortie.txt 2>&1
. cmd 1 2>&1 1> sortie.txt

. cmd1 [1]> sortie.txt [2]>&1



conclusion : la sortie standard et l'erreur standard sont redirigées vers le fichier "sortie.txt"

. cmd 1 2>&1 1> sortie.txt



exemple:

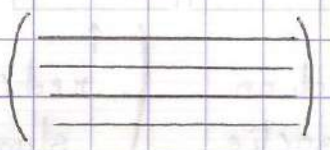
$ ls -l archie 2>&1 1> sortie.txt

↑ fichier n'existe pas

→ affichage erreur sur l'ecran

$ cat sortie.txt
(rien)

$ ls -l 2>&1 1> sortie.txt
$ cat sortie.txt



"listing" du contenue du repertoire courant

5-

\ : neutraliser la signification spécial du caractère ($, *, ?, &, !) qui le suit

exemple :

$ var = 123
$ echo valeur de $var
valeur de 123
$ echo valeur de \$var
valeur de $var

" "

$x = 5
$ echo " La valeur de x est $x "
$ echo " Le repertoire courant est 'pwd' "

La valeur de x est 5
Le répertoire courant est / home / compte 1

=> ça neutralise la signification des caractères et elle permet la substitution des variables et des commandes.

' ' => ça neutralise la signification mais ne sibstitue pas les var et les commandes

ex: $ echo 'la valeur de x est $x'
=> La valeur de x est $x

exercice supplémentaire :
(Script Shell)

écrire un script Shell (NomPrenom.sh) qui demande à l'utilisateur la saisi de son nom et de son prénom et ensuite affiche :
Bonjour Nom Prenom
votre répertoire courant est : /home/compte1
ce répertoire contient les fichiers suivants :
===) contenue du répertoire courant

Réponse :

#! /bin/bash (ou bien /bin/sh)

le chemin vers l'interpreteur (pear - awk ...)

echo "Donner votre nom & prénom"
= il est neutraliser

~~read $nom $prenom~~ read nom prénom
echo "Bonjour $nom $prenom"
↳ le nom de la var pas son contenue !
echo "votre repertoire courant est : `pwd` "
echo "Ce repertoire contient les \ ↵
fichiers suivants : `ls -l` " ↑
neutraliser entrer ⇑

/NomPrenom.sh

~~$ ./NomPrenom.sh~~ erreur (il faux le rendre executable) execution

chmod u+x NomPrenom.sh
$./NomPrenom.sh ou $ /home/compte1/NomPrenom.sh

les droits necessaire pour l'execution d'un script sont la lecture et l'execution

27 - 11 - 2014

Syntaxe du "if"

if [expression 1] (<=> test expression 1)

then

Liste de commandes 1 (si expression 1 est vrai (0))

elif [expression 2]

then

Liste de commandes 2 (si expression 1 est fausse
et expression 2 est vrai)

.
.
.

elif [expression n]

then

Liste de commandes n (si $\forall i = 1, \ldots, n-1$ expression i
est fausse et expression n est vrai)

else

Liste de commandes $n+1$

(si $\forall i = 1, \ldots, n$ expression i fausse et expression $n+1$
est vrai)

fi

notes :

expression peut être une commande !

-e $fich (existance de l'objet $fich)

-d $fich ($fich existe et est un répertoire)

suite de l'exercice 4 :

6 -

```
#!/bin/bash
echo -e "entrez le nom de rep \c"
read REP
if [ ! -d $REP] ; then
    mkdir $REP
fi
cd $REP
echo "On est dans le repertoire"
```

si j'ecris le if et le then sur la même ligne faux les séparé par un point-virgule

---

```
echo "entrez le nom du rep"
read rep
```

ecran

entrez le nom du rep
$Image (clavier)

```
read -p "Entrez le nom du repertoire " rep
```

```
echo -e "entrez le nom du rep \c"
read REP
```

ecran

entrez le nom du rep Image

clavier

Remarque :

$REP = Images

$ mkdir REP → crée un rép avec le nom REP

$ mkdir $REP → crée le répertoire Images

= contenu de la variable

7_

```
# !/bin/bash
echo -e "entrez un nom pour votre fichier"
read fich
if [-e $fich]
then echo "$fich existe"
     chmod go-rwx $fich
else
     echo "$fich n'existe pas"
fi
```

lecture à partir du clavier

./protege Images

$1 argument au script

paramètres de position

script partage

```
# !/bin/bash
if [-e $1]
then
  chmod og-rwx $1
fi
```

la racine de votre compte
(compte de connexion) ~
(répertoire personnel)

~/poubelle

un chemin relatif

$ cd $HOME <=> $ cd ~ <=> $ cd /home/compte1 <=> $ cd

8-

```
#!/bin/bash
read -p "entrez le nom du fich " fich
if [ -f $fich ] ; then
    if [ ! -d ~/poubelle ] ; then
        mkdir ~/poubelle
    fi
mv $fich ~/poubelle
fi
```

le 11_10_2014

Système d'exploitation TP1

installation et utilisation du système Unix

identifiant : compte1
mot de passe : compte1

renomer prof → prof2
SE → SE2

2- ls -l
3- mkdir prof2
ls -l prof2

4- mKdir SE2
5- cd SE2
6- touch smi_a4
7- cat > smi ↵
smi = sci. mathématiques informatique, année scolaire 2014_2015
ctrl D
8- cat smi
9- mkdir TP1
10- ls -l
11- ls -al
12- rm smi_a4
13- mv smi toto
14- cp toto /home/compte1/prof2
15- ls prof2
16- mv toto TP1
17- rmdir TP1
→ le répertoire n'est pas vide !
18- man tar (pour + d'info sur la com tar)

La rép ⇒ tar -cf /home/compte1/prof2/archive.tar SE2/ TP1

tar -tvf /home/compte1/prof2/archive.tar