TE 1
Session unix, introduction à l'arborescence des fichiers

Références
— Polycopié UNIX : chapitres 1, 2, 3, (4)
— Transparents UNIX : sections 1, 2, 3, 4, 5, 6, (9)

Groupes

— A Exercices obligatoires pour les groupes A et B
— AB Exercices obligatoires pour les groupes B et recommandés pour les groupes A
— B Exercices recommandés pour les groupes B et déconseillés pour les groupes A

La mise en place des fichiers d'initialisation de l'UE (exercice 7, p. 7) doit impérativement être faite avant la fin de la première séance. Faire contrôler cette installation par un enseignant.

Machines virtuelles Les ordinateurs de L'UTÈS utilisent un logiciel d'émulation (VMWARE) qui permet de simuler une machine virtuelle fonctionnant sous un des systèmes d'exploitation WINDOWS ou LINUX. L'émulateur propose un menu permettant de choisir la machine virtuelle, menu que l'on retrouvera après l'extinction de la machine virtuelle (état dans lequel on doit laisser le poste en fin de TE).
On utilisera LINUX sous la distribution OPENSUSE : à L'UTÈS pour le moment, cette distribution est installée sans authentification ni persistance. Ainsi, les fichiers créés sur cette machine virtuelle sont perdus à l'extinction.
Pour pallier ces limitations, on utilise l'authentification (via `ssh`) sur le serveur `sappli1` pour accéder à un répertoire personnel sauvegardé et accessible à distance. Ce répertoire peut également être rendu visible sur la machine virtuelle OPENSUSE (par montage `sshfs`).

C'est donc dans ce répertoire personnel, fourni par `sappli1` après authentification, que vous devrez stocker les fichiers que vous souhaitez conserver.

Ex. 1 : Session locale (mode graphique) A 30 min.

Une fois la machine virtuelle OPENSUSE lancée, on propose de commencer par une visite guidée de l'environnement de travail au moyen de l'interface graphique Xfce (Xfce Desktop Environment) [1]. Il existe d'autres environnements de bureau (KDE, Gnome) plus puissants mais aussi plus lourds et gourmands en ressources.
L'environnement Xfce permet d'ouvrir plusieurs fenêtres liées à des applications locales (navigateur internet, éditeur de texte, consoles, calculette...) qui peuvent être lancées sans authentification.

1. Ouvrir une session de travail en choisissant OPENSUSE dans le menu d'accueil de VMWARE. Après quelques dizaines de secondes, le bureau s'affiche (avec une ampoule en fond d'écran) et vous êtes alors connecté en tant qu'utilisateur générique `mnics`.

   (a) Lancer un navigateur et localiser des documentations sur UNIX à l'URL suivant :
   [Polycopié de l'UE] :
   http://wwwens.aero.jussieu.fr/lefrere/master/mni/mni/unix/poly-unix.pdf

1. Noter certaines similitudes entre cette interface graphique et celle de WINDOWS.

(b) Lancer la calculette `kcalc` et modifier la configuration pour qu'elle affiche tous les boutons et toutes les fonctions : repérez notamment l'affichage des entiers en binaire et en octal. Cette calculette servira de convertisseur dans les TE sur les langages.

(c) Ouvrir un des éditeurs de texte[2] accessible via les menus (`kwrite`, `emacs`, `gedit`, `vi`,...) et saisir quelques mots au clavier. Sauvegarder ce texte dans un fichier `toto.txt` sur le répertoire d'accueil `/home/mnics`. Est-il possible de retrouver ce fichier `toto.txt` en se connectant sur un autre poste de travail ?

(d) Retrouver ce fichier à l'aide de l'explorateur graphique de fichiers.

(e) Ouvrir un émulateur[3] de terminal[4] (`xfce4-terminal`, `konsole` ou `xterm` par exemple) et exécuter successivement les commandes :

```
hostname
date
echo  Bienvenue  en  MNI
echo a       b
echo "a       b"
ls
```

Que se passe-t-il si on saisit[5] : `LS` ?

(f) Vérifier que le fichier `toto.txt` est bien présent dans le répertoire d'accueil et visualiser son contenu à l'aide de la commande :

```
cat  toto.txt
```

Que se passe-t-il si on saisit : `cat  TOTO.TXT` ?
Même question pour : `cat toto.txt toto.txt`

(g) Éditer le fichier `toto.txt` en saisissant une commande du type :
`emacs toto.txt` (pour quitter `emacs`, saisir `^X ^C`, c'est-à-dire Ctrl X Ctrl C)
Tester quelques-uns des autres éditeurs disponibles.
▷ **Conseil :** Préciser systématiquement le nom du fichier à éditer dès le lancement de l'éditeur (par exemple `emacs toto.txt` plutôt que `emacs`). Pour cela il faut lancer `emacs` dans l'émulateur de terminal plutôt que via l'interface graphique (cf. 1c). ◁

(h) Fermer le terminal avec la commande `exit`

2. Terminer les applications locales et éteindre la machine virtuelle en **double-cliquant** sur l'icône `Éteindre` de la fenêtre listant les informations de cette session, et qui est apparue dès le chargement de la machine virtuelle OpenSUSE.
L'émulateur VMware doit réafficher le menu initial des machines virtuelles.

3. Relancer la machine OpenSUSE et essayer de retrouver votre fichier `toto.txt` créé lors de la précédente session. Qu'est-il advenu de ce fichier ?

Ex. 2 : Session locale (utilisation de la ligne de commande dans un terminal graphique) A 25 min.

L'exercice 1 a permis d'explorer l'environnement de travail essentiellement en mode graphique. Nous poursuivons ici la visite guidée du système UNIX en tant qu'**interpréteur de commande** (ou `shell`). Fonctionnalités interactives sous le shell `bash` :

copier/coller : boutons gauche, puis milieu de la souris (sans utiliser le menu `Edition` de la console)

effacer un caractère : `Backspace` ou `Suppr` ou encore `Ctrl H`

2. Un éditeur de texte se distingue d'un traitement de texte (word, openoffice writer, libre office writer, LaTeX) par le fait qu'il ne comporte pas d'outils de mise en forme (police, taille des caractères, gras, ...). Le fichier qu'il permet de créer est un fichier texte simple.

3. Un émulateur de terminal est une application qui permet de simuler le comportement d'une console physique en tant que point d'accès à la machine. Par abus de langage, on nommera simplement «terminal» cette application.

4. Les applications émulateur de terminal sont situées dans l'entrée `Système` dans la cascade de menus du bouton `openSUSE` en bas à gauche du bureau. Comme il en sera fait un usage quotidien par la suite, on pourra aussi y accéder via les icônes de raccourcis du tableau de bord pour un accès plus rapide.

5. Les commandes UNIX ne comportent généralement pas de majuscules.

interrompre une commande : `Ctrl C`

historique des commandes : `history` et ↑ ↓ [6]

édition en ligne des commandes : déplacement dans la ligne via ← →

déplacement en début/fin de ligne : `Ctrl A` (début) `Ctrl E` (fin)

complétion (ou complètement automatique pour la saisie rapide des commandes et des noms de fichiers) : `Tab` (tabulation) Cette fonctionnalité permet un gain de temps et évite de nombreuses fautes de frappe. Son usage est plus que recommandé.

Le système UNIX comporte une **documentation en ligne** (commande `man`) pour les commandes qui est d'une aide précieuse. Il faut saisir `q` (quit) pour terminer la consultation du manuel en ligne.

1. Ouvrir un terminal pour travailler en mode ligne de commande. Recréer un fichier `toto.txt` contenant quelques lignes de texte, comme dans l'exercice précédent.
2. Au niveau de l'invite de commande (ou prompt), exécuter successivement les commandes :
   `pwd` (acronyme de print working directory)
   `ls -l` Ne pas confondre `1` (un) et `l` (L)
   `cat toto.txt`
   `xclock`
   En déduire la fonction de chaque commande.
3. Exécuter la commande :
   `man ls`
   et chercher la signification des options `-l` (option format long ou `long format`, utilisée dans la commande `ls -l`) et `-a`.
   Noter que vous pouvez faire défiler page par page avec la barre d'espace.
4. Exécuter les commandes :
   `cp toto.txt titi.txt`
   `ls -l`
   Expliquer l'affichage obtenu et le rôle de commande `cp`.
5. Exécuter successivement les commandes :
   `mkdir mni`
   `ls` puis `ls mni`
   et expliquer le résultat obtenu en s'aidant de la documentation en ligne.
6. Lancer la commande suivante pour télécharger les transparents de cours :
   `wget "http://wwwens.aero.jussieu.fr/lefrere/master/mni/mni/unix/cours-unix.pdf"`
   puis pour visualiser le fichier téléchargé : `evince cours-unix.pdf`
7. Le fichier `toto.txt` est dans le répertoire d'accueil : la commande `ls` permet de le vérifier. La commande `cat toto.txt` permet quant à elle d'afficher son contenu.
   Exécuter successivement les commandes :
   `cp toto.txt mni/`
   `ls mni/`
   `cat mni/toto.txt`
   Commenter les résultats obtenus et expliquer l'effet de la commande `cp`.
8. Exécuter successivement et expliquer le résultat des commandes :
   ```
   cd mni
   pwd
   ls
   mkdir te1
   cp toto.txt te1/tutu.txt
   ls te1
   cd
   pwd
   exit
   ```

6. La navigation dans l'historique (impossible en mode graphique) illustre les atouts du mode ligne de commande.

Ex. 3 : Serveur distant à partir de linux (via Secure Shell) AB 15 min.

Le système UNIX permet de se connecter à des ordinateurs distants (serveurs) au moyen de la commande en ligne `ssh` (ou Secure Shell) qui est un protocole de communication sécurisé.

1. Ouvrir de nouveau un terminal de la machine virtuelle locale OpenSUSE.
2. Ouvrir un deuxième terminal et se connecter au serveur d'applications via `ssh` : remplacer *username* par votre login, c'est-à-dire votre numéro d'étudiant, dans la commande suivante
   `ssh` *username*`@sappli1.datacenter.dsi.upmc.fr`
   puis fournir le mot de passe pour s'authentifier [7] sur le serveur d'applications.
3. Exécuter successivement les commandes ci-dessous dans le terminal local et dans celui connecté au serveur ; noter et comparer les résultats obtenus ; commenter les différences :
   `hostname`
   `uname -a`
   `uname -p`
   `whoami`
   `id`
   `w` (observer le champ `FROM`)
   `who`
   `pwd`
   `ls` (repérer vos fichiers : sur quelle machine sont-ils stockés ?)
   `ls mni`
   `ls /opt` (comparer avec la machine virtuelle locale)
4. Lancer la commande `xclock`. Quel est l'effet de cette commande. Vous pouvez interrompre le processus lancé avec `Ctrl C`.
5. Se déconnecter du serveur d'application avec la commande `logout`.
   Se reconnecter par `ssh` en ajoutant l'option `-x` qui désactive l'affichage des applications graphiques [8] :
   `ssh -x` *username*`@sappli1.datacenter.dsi.upmc.fr`
   Lancer à nouveau la commande `xclock`. Quelle différence observez-vous par rapport à la connexion précédente ?

Ex. 4 : Serveur distant à partir de Windows AB 15 min.

Sous Windows, plusieurs outils permettent de se connecter à des serveurs distants parmi lesquels `Xming` ou `putty` qui est disponible à l'UTES.

1. Fermer la session linux, arrêter la machine virtuelle linux et lancer une session Windows authentifiée.
2. Dans le menu `Outils`, lancer l'utilitaire `putty` en spécifiant les réglages suivants :

   `Host Name` : `sappli1.datacenter.dsi.upmc.fr`

   `Connection Type` : `SSH`

   `Character Set` : dans la catégorie `Window/Translation`, choisir l'encodage `UTF-8` qui est celui par défaut sur le serveur d'applications linux.
3. Ouvrir la connexion, s'identifier et s'authentifier sur le serveur d'applications linux.
4. Exécuter successivement les commandes suivantes et comparer avec les résultats obtenus lors de la connexion depuis la machine virtuelle :
   `hostname`
   `uname -a`
   `whoami`
   `id`

7. En mode graphique, la saisie du mot de passe provoque l'affichage de symboles permettant de compter les caractères saisis. En mode texte, au contraire, **aucun affichage** n'accompagne la saisie du mot de passe.

8. Par défaut sous l'OpenSUSE de l'UTÈS, ssh ouvre un tunnel destiné au transport sécurisé des graphiques au protocole `X11`. Consulter le manuel en ligne pour plus de détails : `man ssh`

`w` (observer le champ `FROM`)
`who`
`pwd`
`ls` (repérer vos fichiers)
`ls mni`

5. Que se passe-t-il[9] si on lance la commande `xclock` ?
6. Fermer la connexion `putty`, fermer la session WINDOWS, arrêter WINDOWS, relancer la machine virtuelle linux pour y redémarrer une session graphique.

Ex. 5 : Pseudo-authentification et sauvegarde de fichiers | A | 15 min.

Nous avons jusqu'à présent travaillé sur la machine virtuelle OPENSUSE sans être authentifié (utilisateur générique `mnics`). Vous n'avez alors pas directement accès à vos fichiers personnels et l'accès à internet est limité à quelques sites de l'Université dont le site de l'UE.
Nous verrons lors de cet exercice comment s'authentifier depuis une session OPENSUSE pour éviter ces limitations[10].

1. Ouvrir un terminal sur la machine virtuelle OPENSUSE.
2. Exécuter successivement les commandes suivantes et noter les résultats obtenus :
   `pwd`
   `ls`
   `ls /home`
   `ls /home/lefrere`
   `ls /home/lefrere/M1`
   `df -h`
3. Lancer la procédure d'authentification en cliquant sur l'icône de LOGIN présente dans la petite fenêtre grise listant les informations de la session `OpenSUSE`. Cette fenêtre sera appelée «Compagnon d'authentification», et ne se trouve que dans le premier espace de travail[11].
   Entrer votre identifiant (numéro d'étudiant) et votre mot de passe dans les champs correspondants. Si l'authentification a fonctionné, vous devez maintenant voir votre identifiant dans le champ `login` du «Compagnon d'authentification» à la place de `mnics`.
4. Dans le terminal précédemment ouvert, exécuter à nouveau les commandes suivantes :
   `pwd`
   `whoami` (commenter !)
   `ls`
   Vérifier l'apparition d'un nouveau répertoire et noter son chemin absolu. Pour comprendre son origine, lancer les commandes :
   `host sappli1.datacenter.dsi.upmc.fr` (adresse du serveur)
   `df -h` (répertoires montés et points de montage locaux)
   Ce nom devrait être composé de votre identifiant et de votre nom de famille
   `ls /home/lefrere/M1` (comparer avec le cas non-authentifié)
5. Créer de nouveau un fichier texte `titi.txt` dans le répertoire d'accueil `/home/mnics`, de la même manière qu'à l'exercice 1. Verifier sa présence et son contenu en utilisant les commandes `ls` et `cat`. Créer également un répertoire `test` directement sur le bureau, soit avec le gestionnaire graphique (clic droit, «nouveau dossier»), soit par la commande `mkdir Bureau/test`.
6. Exécuter dans le terminal les commandes suivantes :
   `cd` ***`n_etudiant`***

9. Pour lancer des applications graphiques via `putty`, il faut disposer d'un serveur X11 sur la machine windows : plusieurs outils sont installables, notamment `MobaXterm` et `Xming`. Consulter par exemple sur la page de l'UE :
http://wwwens.aero.jussieu.fr/lefrere/master/mni/mni/unix/GuideMobaXterm.pdf ou
http://wwwens.aero.jussieu.fr/lefrere/master/mni/mni/unix/xming.pdf

10. Notez que ceci n'est pas une limitation intrinsèque aux machines virtuelles Linux, mais un choix du service informatique

11. Quatre espaces de travail sont disponibles. Vous pouvez passer de l'un à l'autre, soit en cliquant sur leur icône (un carré gris foncé) à droite du tableau de bord, soit par les raccourcis clavier `Ctrl Alt ←` et `Ctrl Alt →`

n_etudiant est le point de montage de votre «répertoire personnel» dont vous avez dû constater l'apparition après l'authentification. La complétion avec Tab permet de ne saisir que les premiers chiffres de votre numéro d'étudiant.

```
pwd
ls
mkdir mni
ls -l
mkdir mni/te1
```

7. Créer un second fichier de texte `toto.txt` contenant quelques lignes, dans le sous-répertoire `te1`, par exemple avec la commande :
   `gedit mni/te1/toto.txt` (ou avec l'éditeur de texte de votre choix)
8. Se connecter au serveur d'application `sappli1`. Pour ce faire, à condition d'être authentifié sur la machine locale OpenSUSE, vous pouvez utiliser le raccourci (ou l'alias) : `ssh sappli1`
   Exécuter alors les commandes, depuis le serveur distant :
   `whoami`
   `pwd`
   `xclock` Le raccourci utilise-t-il l'option `-X` ?
   `ls -l` Vérifier la présence du répertoire `mni` ; quelle est sa date de création ?

   Chercher le fichier `toto.txt` créé précédemment, et vérifier son contenu avec la commande `cat`. Chercher également le fichier `titi.txt` depuis le serveur distant. Qu'en déduire sur l'emplacement physique de stockage des fichiers `titi.txt` et `toto.txt`.
9. Se déconnecter du serveur d'application par la commande `logout`. Éteindre et redémarrer la machine OpenSUSE. Rechercher les fichiers `titi.txt` et `toto.txt`, ainsi que le répertoire `test`. Conclure sur la persistance des fichiers et des répertoires créés lors d'une session OpenSUSE, en fonction de leur position dans la hiérarchie des répertoires.

Pour pouvoir retrouver vos fichiers d'un TE à l'autre, il faut vous authentifier et tout enregistrer sous votre répertoire personnel.

À partir de cet exercice et pour tous les TE suivants, on s'authentifiera sytématiquement dès le début de la session `OpenSUSE`. De plus, on stockera tous les fichiers créés sous le répertoire partagé avec le serveur, sous peine de les perdre.

Ex. 6 : Exploration du compte de l'UE : arborescence de fichiers A 15 min.

Le but de cet exercice est de commencer à se familiariser avec la structure de l'espace de travail. En effet, sous UNIX, l'ensemble des fichiers est structuré sous la forme d'une hiérarchie de répertoires et de fichiers constituant un arbre unique.

▷ **Conseil :** Utiliser la complétion Tab pour saisir rapidement (et correctement) ces commandes ! ◁

1. Afin de se repérer dans la hiérarchie du système de fichiers, exécuter les commandes qui suivent en précisant, pour chaque question, où l'on se trouve dans l'arborescence. Au fur et à mesure de l'exploration de cette hiérarchie de fichiers, construire sa représentation graphique sous forme d'arbre ;
   (a) `cd` puis `pwd` puis `ls`
   (b) `cd /` puis `pwd` puis `ls`
   (c) `cd /home/lefrere` puis `pwd` puis `ls`
   (d) `cd /home/lefrere/M1` puis `pwd` puis `ls`
   (e) `cd Doc/f90+c` puis `pwd` puis `ls`
   (f) `cd fortran` puis `ls`
   (g) `cd ../../unix` puis `ls`
   (h) `cd` puis `ls`

2. Afficher [12] le polycopié de cours UNIX avec une des commandes suivantes :
   `evince /home/lefrere/M1/Doc/unix/poly-unix/poly-unix.pdf &`
   `gv /home/lefrere/M1/Doc/unix/poly-unix/poly-unix.ps &`

Ex. 7 : Installation des fichiers d'initialisation de l'UE **A** **15 min.**

Il est indispensable de personnaliser l'environnement de travail afin de faciliter l'interaction avec le shell et de définir des paramètres communs utiles pour ces travaux encadrés. Cette opération peut être réalisée simplement en modifiant les fichiers personnels d'initialisation des sessions (`.bash_profile` et `.bashrc`), mais il s'agit de fichiers très « sensibles ». On préferera donc les remplacer [13] par les fichiers d'initialisation mis à disposition dans le répertoire `M1/Config/` du compte de référence `/home/lefrere` [14] de l'UE. C'est sur ce compte que seront mis à disposition des documentations, des exemples de programmes et des fichiers nécessaires pour les TE.

1. Ouvrir un terminal et **se connecter au serveur d'application** `sappli1`.
2. Exécuter les commandes suivantes en respectant les « points », le tilde [15] «`~`» et les « espaces » :

`cd`	retour au répertoire d'accueil
`ls -al`	liste avec fichiers cachés
`cp .bashrc .bashrc.000`	sauvegarde des fichiers initiaux
`cp .bash_profile .bash_profile.000`	par précaution

   puis la copie proprement dite en **confirmant** [16] l'autorisation d'écraser les fichiers initiaux :
   `cp /home/lefrere/M1/Config/etudiants.bash_profile .bash_profile`
   `cp /home/lefrere/M1/Config/etudiants.bashrc .bashrc`

   et enfin afficher les nouveaux fichiers pour information :
   `cat .bash_profile`
   `cat .bashrc`
3. Sans fermer la fenêtre active (seul moyen de récupérer le compte en cas d'erreur grave dans les fichiers d'initialisation), ouvrir une nouvelle fenêtre terminal grâce à la commande :

   `xterm -ls &` l'option `-ls` (login shell) permet l'exécution des scripts d'ouverture de session [17]

   Le nouvel environnement doit être automatiquement activé dans ce nouveau terminal. Vérifier quelques fonctionnalités installées grâce à ces fichiers de configuration :

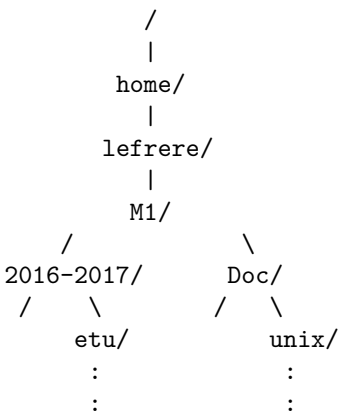
   — invite personnalisée (avec un retour ligne),

   — alias imposant des options aux compilateurs `gcc` et `gfortran` mis en place ; les afficher avec la commande interne (builtin) `alias`.

   Parmis les alias définis sur le système, certains sont propres au système d'exploitation. C'est le cas de `cp` et `rm`, alias de `cp -i` et `rm -i` (`interactive`), qui demande confirmation avant de supprimer des données. Ces alias peuvent ne pas être définis sur d'autres systèmes, `cp` et `rm` peuvent donc y avoir un comportement dangereux.
   On peut alors fermer la console initiale.
4. Ouvrir un nouveau terminal local. Vérifier que l'environnement personnalisé pour l'UE est maintenant systématiquement pris en compte dans tous les terminaux.

12. Attention : ne pas imprimer le polycopié qui est distribué. Cela épuiserait votre quota de pages et saturerait les imprimantes du centre.
13. En fait, on les remplacera par des fichiers qui eux-mêmes sont chargés de lancer l'exécution de ceux du compte de l'UE. Cette méthode permet de suivre automatiquement les mises à jour.
14. Attention à ne pas confondre `1` (un) et `l` (L).
15. Sur les claviers français, le tilde est obtenu en appuyant simultanément sur les touches `Alt Gr` et `é`. D'autres combinaisons de touches utilisant le préfixe `Alt Gr` seront utiles sous UNIX notamment pour obtenir le tube (pipe) `|`, l'arrobase `@`, les accolades `{` et `}`, les crochets `[` et `]` ainsi que la contre-oblique (backslash) `\`.
16. Lire le message et répondre : ne pas se contenter de saisir `Entrée` !
17. Si on préfère lancer `konsole`, il faudra le configurer pour exécuter la commande `/bin/bash -l`

Ex. 8 : Analyse d'une arborescence A 15 min.



1. En utilisant les commandes `cd`, `pwd` et `ls`, compléter le schéma des arborescences `etu` et `unix` du compte de `lefrere` à partir des pointillés de la figure ci-contre. On se limitera aux trois premiers niveaux sous `etu`.
2. Vérifiez votre arbre avec la commande `ls -R`.
3. Choisir comme répertoire de travail celui de l'UE MNCS de l'an dernier, c'est-à-dire `/home/lefrere/M1/2016-2017/etu/mncs`. On souhaite lister le contenu de `te1` (de l'an dernier) situé dans le répertoire `te` de `unix` dans `mni` sans changer de répertoire de travail. Compléter la commande `ls ../`
4. Refaire cette liste en utilisant un chemin absolu.
5. Se placer dans le répertoire `te1` de la hiérarchie de `2016-2017`. On souhaite lister le contenu de `Doc/unix/poly-unix`. Compléter la commande `ls ./`

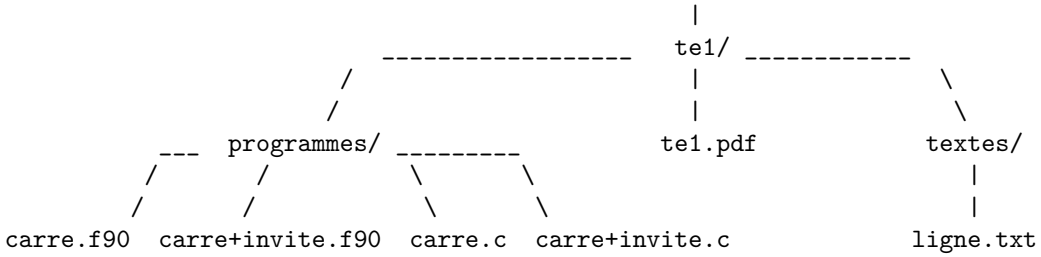
Ex. 9 : Copies d'un fichier AB 5 min.

L'objectif est maintenant de copier l'énoncé du TE1 de cette année (`te1.pdf`) situé dans le répertoire `/home/lefrere/M1/2017-2018/etu/mni/unix/te/te1` dans **votre** répertoire `te1` (créé dans la question 6 de l'exercice 5). On effectuera la copie dans chacune des trois situations suivantes et on numérotera `a`, `b` et `c` les copies. Indiquer la commande à lancer dans chaque situation.

1. Choisir `/home/lefrere/M1/2017-2018/etu/mni/unix/te/te1` comme répertoire de travail et utiliser un chemin relatif pour la source et un chemin absolu pour la destination (`copie-a.pdf`)
2. Choisir `/home/mnics/`*`n_etudiant`*`/mni/te1` comme répertoire de travail et utiliser un chemin absolu pour la source et un chemin relatif pour la destination (`copie-b.pdf`)
3. Choisir le répertoire d'accueil de `mnics` comme répertoire de travail et n'utiliser que des chemins absolus (`copie-c.pdf`).

Ex. 10 : Création d'une arborescence AB 35 min.

1. Sous votre répertoire `mni`, créer un répertoire `unix` et un nouveau sous-répertoire `te1` [18].
2. Créer dans ce nouveau répertoire `te1` une arborescence de répertoires et y déposer des copies des fichiers disponibles dans le répertoire `te1` de `/home/lefrere/M1` de façon à aboutir à la structure suivante :



   Vérifier l'arborescence ainsi construite en affichant la liste récursive des fichiers à partir de `te1`.
3. Choisir `programmes` comme répertoire de travail. Créer, dans le répertoire `programmes`, deux sous-répertoires `fortran` et `C` et déplacer les quatre fichiers sources selon le langage indiqué par leur suffixe.
4. Vérifier l'arborescence avec les outils `tree` et `ls -R`.

18. Il est donc distinct de `te1` créé précédemment directement sous `mni`.

Ex. 11 : Caractères jokers et options de la commande ls **AB** **45 min.**

A

1. Choisir votre répertoire d'accueil comme répertoire de travail. Saisir les commandes ci-contre. Indiquer si la commande liste un répertoire ou un fichier ordinaire. Noter l'effet des différentes options, et le fait que l'on peut combiner les options.

```
ls
ls -a
ls -l
ls -al
ls -dl
ls -lh
ls -R
ls /usr/bin
ls -lh /usr/bin/fold
```

AB

2. Ouvrir un nouveau terminal ; y lancer la commande : `unalias ls` [a]
Tester ensuite et expliquer le résultat des commandes ci-contre (analyser en détail le résultat des deux dernières commandes). Fermer ce terminal à la fin de la question.

```
ls /bin
cd /bin
ls
ls m*
ls m[a-o]*
ls m[!a-o]*
ls *dir
ls [lm]*
ls [!lm]*
```

```
cd /usr/bin
ls lp*
ls lp?
ls lp??
ls k*.*
cd /home/lefrere/M1/Doc
ls
ls *
ls */*
```

a. Sur certains sytèmes linux, `ls` est en fait l'alias de `ls --color=auto`. Chaque appel de `ls` par l'utilisateur lance en réalité la commande `ls` avec l'option `--color=auto`. La commande interne `unalias` supprime l'alias dans le terminal où elle est saisie et permet de retrouver le comportement naturel de `ls`.

3. Choisir le répertoire `/bin` comme répertoire de travail. Quelle commande doit-on saisir pour faire la liste de tous les fichiers et répertoires dont le nom contient 3 caractères exactement ? dont le nom contient 6 caractères au moins ?

B

4. Ouvrir un nouveau terminal et y lancer les commandes ci-dessous. Quels sont les effets de l'alias sur l'affichage de la liste ? Fermer le terminal à la fin de la question.

```
ls
type ls
\ls
unalias ls
type ls
ls
ls -F
ls --color=auto
```

Ex. 12 : Autres commandes **B**

Cet exercice facultatif propose de découvrir quelques commandes supplémentaires non essentielles pour la suite. Exécuter les commandes listées ci-dessous et commenter le résultat de leur exécution (s'aider de la documentation avec la commande `man`).

1. Diagnostics système
`top` [`q` pour sortir]
`du` [`-h` pour afficher les tailles en octets, ko, Mo,...]
`du --max-depth=1`
2. Amusant et pratique
`cal`
`yes` [`Ctrl C` pour en sortir... essayer également `yes no`]

Ex. 13 : Travail à la maison A

Afin de pouvoir préparer les TE et vous entraîner en dehors des séances, nous vous demandons de mettre en place dès maintenant des outils pour travailler depuis chez vous. Plusieurs solutions plus on moins simples à installer sont envisageables selon la configuration dont vous disposez.

1. UNIX (Mac ou linux) : Vous pouvez utiliser la console de votre système qui à quelques détails près donnera les mêmes résultats que les machines de L'UTÈS.
2. WINDOWS : De nombreuses solutions existent de complexités diverses. Nous vous en présentons quelques-unes en partant de la plus simple. Les outils indiqués par une étoile sont documentés sur la page de l'UE : http://wwwens.aero.jussieu.fr/lefrere/master/mni
   - Le logiciel de connexion à distance `MobaXterm` (∗) est très simple d'installation et assure les fonctions essentielles.
   - Le logiciel d'émulation graphique `Xming` (∗) peut aussi permettre la connexion à distance.
   - Signalons aussi l'installation de linux sur clé usb (voir http://www.linuxliveusb.com/) ou sur machine virtuelle.

   Quelle que soit la solution choisie, vous devrez la tester dès la première semaine, pour pouvoir travailler les 3 TEs de la partie unix.

TE 2
Compléments sur la hiérarchie de fichiers – Redirections

Références

— Polycopié UNIX : chapitres 2, 3, 4, 5,10

— Transparents UNIX : sections 3, 4, 5, 6, 7, 8, 9, 10, 11

Créer, dans votre répertoire `unix`, un sous-répertoire `te2` pour y placer tous les fichiers du TE 2.

Ex. 1 : Droits d'accès **AB**

Droits sur les fichiers ordinaires **A** | **10 min.**

1. Afficher le fichier `/home/lefrere/M1/Config/etudiants.bash_profile`, puis afficher les attributs de ce fichier (droits, propriétaire, taille, date, ...). Repérer quels droits d'accès font que vous pouvez le lire, mais que vous ne pouvez pas écrire dans ce fichier.
2. Afficher les droits du fichier de la commande `ls` (fichier `/bin/ls`).
3. Choisir comme répertoire de travail le répertoire `C` créé (sous `programmes`) dans l'exercice 10 du TE 1. Compiler le fichier `carre+invite.c` avec la commande
   `gcc-mni-c89 carre+invite.c`
   Cela produit un fichier `a.out` que vous pouvez exécuter avec la commande
   `./a.out`
   Repérer les droits de ce fichier. Les modifier afin que vous seul puissiez exécuter ce fichier.
4. Protéger le fichier `carre+invite.c` afin que personne (ni même vous) ne puisse le lire. Vérifier en essayant de l'afficher.

Droits sur les répertoires **B** | **20 min.**

1. Quels droits du répertoire `programmes` permettent de faire la liste des fichiers qu'il contient ? Les modifier temporairement et vérifier l'effet produit, puis redonner ces droits.
2. Quels droits du répertoire `programmes` permettent de le traverser pour atteindre un fichier plus bas dans la hiérarchie ? Les modifier temporairement et vérifier l'effet produit, puis redonner ces droits.
3. Quels droits du répertoire `programmes` permettent de copier ou détruire des fichiers dans ce répertoire ? Les modifier temporairement et vérifier l'effet produit en essayant d'y copier le fichier `ligne.txt`, puis redonner ces droits.
4. Exécuter les commandes suivantes et expliquer.
   ```
   chmod a-r,a+x programmes
   ls programmes
   cd programmes/C
   ls
   ```
5. Créer dans votre répertoire d'accueil un (sous-)répertoire de nom `bin`. Copier le fichier `a.out` dans `bin`. Protéger `bin` afin que personne d'autre que vous ne puisse y accéder.

Ex. 2 : Recherche de fichiers **AB** | **25 min.**

AB

1. En utilisant la commande `find`, chercher sous `/home/lefrere/M1/` tous les fichiers dont le nom est suffixé par `.c` (fichiers sources de programmes en C).

2. Utiliser `find` pour établir la liste de tous les répertoires situés sous votre répertoire d'accueil. Que remarquez-vous ?

B

3. Afficher la liste de tous les fichiers de taille inférieure à 1000 octets dans votre répertoire d'accueil et ses sous-répertoires. Vérifier que le critère de sélection est bien respecté. Comment procéder pour afficher leurs chemins absolus ?

Ex. 3 : Archivage et compression AB 25 min.

A

1. L'objectif est de créer dans votre répertoire `unix` un fichier `doc.tar` qui archive la branche `Doc` de la hiérarchie située sous le répertoire `/home/lefrere/M1/`. On choisit donc le répertoire `/home/lefrere/M1/` comme répertoire de travail. Quelle est la taille du fichier d'archive en octets ?
2. Vérifier la structure de l'arborescence ainsi stockée dans `doc.tar`.
3. Reconstruire à partir de l'archive cette arborescence de fichiers sous votre répertoire `te2`.
4. Parcourir l'arborescence ainsi restituée et y localiser notamment le fichier `poly-unix.pdf`.

B

5. Déplacer l'arborescence installée (`Doc`) dans votre répertoire `unix`.
6. Compresser le fichier d'archive. Quel est le taux de compression ?
7. Vous envoyer le fichier d'archive compressé en fichier attaché par mail.
8. Recommencer l'opération 1 en compressant l'archive lors de sa création (suggestion : `man tar`).
9. Reprendre l'opération de compression avec `bzip2` (éventuellement avec `unlzma` ou `xz`). Comparer le taux de compression avec celui de `gzip`.

Ex. 4 : Téléchargement d'une arborescence et extraction B 20 min.

L'objectif est d'installer sur votre compte une hiérarchie de travaux pratiques du cours d'unix de l'IDRIS, disponible sous la forme de fichier d'archive compressée à l'URL
`"http://www.idris.fr/media/formations/utilisation-unix/tp.tar.gz"`

1. Pour accéder à l'extérieur de l'UPMC, se connecter au serveur `sappli1`. Choisir `~/tmp/` comme répertoire de travail. avec la commande `wget` Télécharger le fichier `tp.tar.gz` Revenir ensuite sur la machine locale et vérifier où se trouve le fichier téléchargé. Lancer les opérations suivantes sur la machine locale.
2. Quelle est la taille du fichier téléchargé ? Décompresser le fichier. Quel était le taux de compression ?
3. Vérifier le contenu de l'archive avant d'en extraire l'arborescence.
4. Extraire les fichiers de l'archive de façon à ce que le répertoire `TP` soit un sous répertoire de votre répertoire `te2`. Dessiner l'arborescence restituée.
5. Détruire tous les fichiers de cette hiérarchie dont le nom commence par `mpp`
6. Archiver la nouvelle hiérarchie dans un fichier `TP-IDRIS.tar` placé dans votre répertoire d'accueil.
7. Détruire l'arborescence modifiée `TP`
8. Restituer l'arborescence modifiée à partir de l'archive `TP-IDRIS.tar`

Ex. 5 : Sortie standard, sortie d'erreur standard et redirections **A** **15 min.**

1. Depuis le répertoire `/home/lefrere/M1/2016-2017/etu/mni/unix/te/te2/`, copier les fichiers `debut` `fin` et `redir.sh` dans votre répertoire `te2`
2. Exécuter les commandes suivantes et expliquer le résultat :
   `cat debut`
   `cat debut fin`
   `cat debut fin > sortie` puis `less sortie`
   `cat debut /inexistant`
   `cat debut /inexistant > sortie` puis `less sortie`
   `cat debut /inexistant 2> error` puis `less error`
   `cat debut /inexistant > sortie 2> error` puis `less sortie` et `less error`
3. Afficher le fichier texte `redir.sh` : il comporte des commandes, c'est un shell-script.
   Essayer d'exécuter ce script avec la commande `./redir.sh` Observer l'affichage : que constatez-vous ? Ajuster les droits du fichier `redir.sh` et recommencer.
4. Rediriger la sortie de ce script vers un fichier `memo.txt` Vérifier le résultat en affichant `memo.txt`
5. Sans passer par un éditeur de texte, ajouter le message "Fin du memo" à la fin de `memo.txt`.

Ex. 6 : Utilisation du filtre `tr` **AB** **15 min.**

Recopier dans votre répertoire de travail le fichier
`/home/lefrere/M1/2016-2017/etu/mni/unix/te/te2/ligne-ascii.txt` Puis tester le comportement des commandes suivantes ; noter les résultats et les expliquer en s'aidant du manuel en ligne de `tr` :

```
cat ligne-ascii.txt
tr aeiou AEIOU  <  ligne-ascii.txt
tr "aeiou" 'AEIOU'  <  ligne-ascii.txt
tr aeiou AEIOU  <  ligne-ascii.txt  >  LIGNE-ascii.txt
cat LIGNE-ascii.txt
cat ligne-ascii.txt  |  tr aeiou AEIOU
tr '0123456789' 'abcdefghij'  <  ligne-ascii.txt
tr '0-9' 'a-j'  <  ligne-ascii.txt
tr "[:lower:]" "[:upper:]"  <  ligne-ascii.txt
tr "a-z" "A-Z"  <  ligne-ascii.txt
tr '0-9' '+'  <  ligne-ascii.txt

tr -d 'a-d' < ligne-ascii.txt
echo '1a2aa3aaa4bbbb' | tr 'a' 'A'
echo '1a2aa3aaa4bbbb' | tr -s 'a' 'A'
echo '1a2aa3aaa4bbbb' | tr -s 'a' '\n'
tr '0-4' '+=' < ligne-ascii.txt
tr -s '0-4' '+=' < ligne-ascii.txt
```

Ex. 7 : Filtres élémentaires `head`, `tail`, `wc`, `tr` et tubes **A** **15 min.**

1. Rechercher le fichier `comptes-lutes.txt` situé quelque part sous `Doc` sur le compte de l'UE. Le recopier dans votre répertoire `te2`
2. Afficher le nombre de lignes de ce fichier.
3. Afficher le nombre d'octets de ce fichier. Afficher le nombre de caractères de ce fichier. Comporte-il des caractères accentués ?
4. Afficher sa première ligne, puis l'afficher en lettres capitales. Stocker le résultat dans un fichier appelé `Titre`.
5. Afficher les lignes 23 à 26 de ce fichier. Afficher le nombre de mots qu'elles comportent.

Ex. 8 : Programmes et redirections A 20 min.

Les programmes `carre.f90` en fortran et `carre.c` en C (fournis au TE2) lisent trois nombres fournis au clavier [1], calculent leur carré, puis affichent ces carrés.

1. Compiler le programme fortran (`gfortran-mni carre.f90`) puis tester le fonctionnement de l'exécutable `a.out` ainsi produit. Renommer le fichier exécutable en `carre-f.exe`.
2. Compiler le programme C (`gcc-mni-c89 carre.c`) et tester le fonctionnement du nouvel exécutable `a.out`. Renommer le fichier exécutable en `carre-c.exe`.
3. Les filtres du système unix ne sont pas les seuls à pouvoir utiliser les redirections : les programmes qui lisent au clavier et écrivent à l'écran acceptent aussi les redirections.

   Saisir trois nombres dans un fichier que l'on nommera `in` et faire afficher leurs carrés.
4. Comment stocker les résultats de plusieurs exécutions successives du programme dans un fichier `out` ? Où est envoyé le message d'erreur dans le cas d'une saisie non numérique avec l'exécutable fortran ? Comment le stocker dans un fichier nommé `erreur` ?
5. AB Comment calculer les puissances quatrièmes des entrées sans modifier le programme source ?
6. B Comment récupérer aussi les carrés dans un fichier ?

Ex. 9 : Utilisation de l'éditeur `vi` AB

1. À partir du répertoire : `/home/lefrere/M1/2016-2017/etu/mni/unix/te/te2/`, recopier dans le répertoire de travail que vous vous êtes créé pour cette séance les fichiers :

   `ref-vi-vim.txt` qui est le fichier à éditer

   `instructions-vi-vim.txt` qui contient l'exercice à effectuer et qu'il est conseillé d'imprimer.
2. Ouvrir le fichier `ref-vi-vim.txt` sous `vi` et appliquer les actions décrites dans le fichier texte `instructions-vi-vim.txt` (ce qui suit le caractère `#`, et le caractère `#` lui-même, ne doivent pas être entrés au clavier). Commenter l'effet des requêtes proposées.

1. Attention : contrairement à `carre+invite.{c,f90}`, ces programmes n'affichent aucun message invitant à saisir les données d'entrée.

TE 3

Redirections, tubes et filtres

Références

— Polycopié UNIX : chapitres 5, 6, 7, 8, 9, **10**

— Transparents UNIX : sections **10**, 11, 12, 13, 14, 15

Créer un répertoire `te3` sous votre répertoire `unix` : on y placera tous les fichiers de travail du TE3. En particulier, on y copiera tous les fichiers utiles fournis dans le répertoire `te3` du compte de l'UE.

Ex. 1 : Tubes et filtres (`more, less, wc, head, tail, sort, grep`) A 25 min.

Pour chaque question de 1 à 7, utiliser la commande `\ls` [1] avec des options éventuelles, puis un filtre à l'aide d'un tube.

1. Afficher la liste des fichiers du répertoire `/usr/bin/` en contrôlant le défilement écran par écran (tester avec `more` puis avec `less`).
2. Compter le nombre de fichiers de ce répertoire (`wc`).
3. Afficher les noms des 8 derniers fichiers de `/usr/bin/` (`tail`)
4. Afficher les noms des 8 premiers fichiers de `/usr/bin/` (`head`)
5. Afficher les noms des fichiers de `/usr/bin/` classés par ordre alphabétique inverse (`sort`).
6. Compter le nombre des fichiers du répertoire `/home/lefrere/M1/Config/`
7. Afficher la liste des fichiers du répertoire `/home/lefrere/M1/Config/` avec leurs attributs ; les classer par ordre de taille décroissante. Compter le nombre de fichiers classés. Pourquoi ne retrouve-t'on pas le nombre précédent ? Insérer un filtre pour éviter ce problème.

On s'intéresse maintenant à des fichiers d'entête pour le préprocesseur `cpp`

8. Afficher les lignes du fichier `/usr/include/limits.h` qui comportent la chaîne `MAX` (`grep`). Compter leur nombre.
9. Afficher les 3 premières lignes du fichier `/usr/include/limits.h` qui comportent la chaîne `MAX`.
10. AB Afficher les lignes du fichier `/usr/include/limits.h` qui comportent à la fois l'instruction préprocesseur `#define` en début de ligne (avec un nombre arbitraire de blancs entre `#` et `define`) et la chaîne `MAX`

Ex. 2 : Fichier de notes avec `awk` A 15 min.

Le fichier `notes.txt` contient les notes d'un groupe d'étudiants, précédées d'une ligne d'entête.

notes.txt

```
Nom Prenom Initiales cc1 cc2 examen oral 2011-2012
Legrand Jean JL 10 12 13 14
Dupont Yves YD 9 7 9 12
Durand Claire CD 8 12 11 10
Lefort Corine CL 15 14 14 12
Duran Alberto AD 14 13 11 09
```

Utiliser le filtre `awk` pour :

1. Afficher, pour chaque ligne du fichier `notes.txt`, le numéro de la ligne, le premier champ, le nombre de champs et le dernier champ. Afficher le nombre de lignes du fichier `notes.txt`

1. La commande `\ls` invoque la commande native `/bin/ls` au lieu de l'alias qui pourrait, selon les configurations de site, utiliser la colorisation automatique et, éventuellement, l'affichage des caractères `*`, `@` ou `/` en fin de nom pour respectivement, les fichiers exécutables, les liens et les répertoires.

2. Afficher le fichier `notes.txt` en permutant les champs `Nom` et `Prenom`.
3. Extraire du fichier `notes.txt` les lignes des étudiants sans la ligne d'entête et stocker le résultat dans le fichier `notes.dat` avec lequel on travaillera par la suite.
4. Calculer la moyenne de l'épreuve d'oral pour l'ensemble du groupe d'étudiants et l'afficher.
5. Calculer la moyenne des notes de l'étudiant dont les initiales sont `YD` et l'afficher.
6. Afficher la liste des étudiants ayant obtenu une note comprise entre 10 et 14 (inclus) au premier contrôle continu et calculer la moyenne de ces notes.

Ex. 3 : Analyse de fichier image en niveaux de gris **AB** | **40 min.**

Les fichiers d'image de type `pgm` en niveaux de gris (**p**ortable **g**ray **m**ap) peuvent être stockés sous les formats `P2` (ascii) ou `P5` (binaire). Mais ils comportent tous un en-tête de trois lignes en texte ascii décrivant successivement :

ligne 1) le format de fichier sous la forme `P2` dans le cas `pgm` ascii par exemple
ligne 2) le nombre de colonnes (largeur) et le nombre de lignes (hauteur) de l'image
ligne 3) la valeur maximale du niveau de gris qui représente le blanc

Suivent alors les valeurs des pixels en niveau de gris stockées en ascii pour un fichier de type `P2`. L'image est parcourue de gauche à droite et de haut en bas ; le nombre de pixels par ligne du fichier ascii ne détermine pas la géométrie de l'image, qui est spécifiée dans l'en-tête (`man pgm` pour plus de détails).
Le fichier `img-asc.pgm`, fourni sur le compte de l'UE, est un fichier ascii dans lequel les niveaux de gris sont des entiers compris entre 0 et 255 (blanc).

premières lignes du fichier img-asc.pgm

```
1 P2
2 92 124
3 255
4 158 125 133 111 130 140 124 232 255 224 254 255 226 230 226 222 208 194 179 197 144 174 171 162 153 155
5 160 157 158 155 165 173 170 207 237 233 223 216 216 236 250 255 252 242 255 255 248 252 247 253 251 244
6 248 255 254 242 246 242 255 247 255 240 254 255 242 255 255 244 239 214 230 195 223 197 207 202 236 204
7 230 199 238 255 204 172 185 162 217 136 79 57 89 113
8 162 66 102 93 179 188 148 192 177 242 209 198 211 164 225 168 218 131 133 142 182 153 170 144 152 134
```

Travail sur le fichier `img-asc.pgm` **A**

1. Quelle commande affiche le nombre total de lignes du fichier (et non de l'image) `img-asc.pgm` ?
2. Quelle commande permet d'extraire les trois premières lignes du fichier `img-asc.pgm` et de les écrire dans un fichier `entete.txt` ?
3. Extraire la partie du fichier `img-asc.pgm` qui suit l'en-tête (soit de la ligne 4 à la fin), pour former le fichier `pixels.txt` sur lequel on travaillera dorénavant.

Travail sur le fichier `pixels.txt`

1. [A] Quelle commande permet de compter le nombre de pixels de l'image ? Avec quelle commande vérifier à partir de l'en-tête ?
2. [AB] À l'aide du filtre `tr`, écrire les valeurs des niveaux de gris des pixels à raison d'un par ligne dans le fichier `listpix.txt`
3. [B] Reprendre les deux questions précédentes en utilisant `awk` sur le fichier `pixels.txt`.

Travail sur le fichier `listpix.txt` **AB**

1. Quelle commande permet de vérifier le nombre total de pixels dans le fichier `listpix.txt` ?
2. Comment compter le nombre de pixels dont le niveau de gris est `200` dans le fichier `listpix.txt` ? Même question avec le niveau `20`.
3. Comment afficher les valeurs des niveaux de gris par ordre décroissant ?
4. Comment afficher la valeur du niveau de gris du pixel le plus clair ? le plus foncé ?
5. Calculer le niveau de gris moyen de l'image.

Ex. 4 : Image bitmap (éditeur interactif, `sed` et `tr`) **AB** **20 min.**

Le fichier texte `esperluette.pbm` est une image du caractère & en noir sur fond blanc au format `pbm` (**p**ortable **b**it**m**ap) ascii, semblable à celui des images en niveaux de gris de l'exercice 3 :

— les deux premières lignes de ce fichier sont un en-tête que l'on ne modifiera pas : la première indique le type de fichier (`P1` pour `pbm` ascii), la seconde la géométrie de l'image (nombre de lignes et de colonnes) ;

— les lignes suivantes correspondent aux pixels noirs (1) ou blancs (0) de l'image parcourue de gauche à droite puis de haut en bas ; chaque ligne comporte au plus 70 caractères.

Copier ce fichier depuis le compte de l'UE. Afficher l'image qu'il représente avec `display` ou `xv` par exemple. L'objectif est de modifier ce fichier texte pour afficher le caractère `&` en blanc sur fond noir, via différentes méthodes :

1. avec un éditeur de texte interactif ;
2. avec le filtre `sed` ;
3. AB avec le filtre `tr` et d'autres filtres élémentaires.

Ex. 5 : Trace d'une matrice carrée (`awk`) **A** **15 min.**

Le fichier `mat5-5.dat` stocke une matrice carrée 5 par 5. Sans faire d'hypothèse sur la taille de la matrice, écrire les programmes `awk` qui calculent :

1. la somme des éléments de sa deuxième colonne ;
2. la somme des éléments de sa troisième ligne ;
3. AB sa trace (somme des éléments diagonaux).

`mat5-5.dat`

```
11 12 13 14 15
21 22 23 24 25
31 32 33 34 35
41 42 43 44 45
51 52 53 54 55
```

Ex. 6 : Édition d'un fichier fortran 90 (`sed`, `grep`) **B** **15 min.**

On considère le fichier fortran 90 `trinome.f90`. Pour répondre aux questions suivantes, utiliser les filtres `grep`, `sed` ou un éditeur en repartant à chaque fois de la version originale de `trinome.f90`.

1. Faire une copie de sauvegarde du fichier `trinome.f90`.
2. Créer un fichier constitué des lignes 23 à 46 de `trinome.f90`.
3. Créer un fichier ne contenant que les lignes de commentaires de `trinome.f90`. On rappelle qu'en fortran 90, une ligne est un commentaire si son premier caractère non-blanc est un point d'exclamation (`!`).
4. Créer un fichier omettant tous les commentaires de `trinome.f90`
5. Ajouter des blancs après les virgules.
6. Insérer des blancs de part et d'autres des opérateurs arithmétiques `"+-"`.
7. Passer toutes les ocurrences des mots clefs `if`, `then`, `else`, et `endif` en majuscules.

Ex. 7 : Mise en forme pour graphique (`awk`) **B** **20 min.**

On souhaite représenter graphiquement l'évolution de l'endettement consigné dans le fichier `depenses`.

1. Calculer l'axe des temps en jours depuis le début de l'année 2000. Pour simplifier, on prendra des mois de 30 jours et des années non bissextiles.
2. Ordonner les emprunts en fonction de ce numéro de jour.
3. Stocker dans un fichier de trois colonnes nommé `dettes.dat`, le numéro du jour, le montant de l'emprunt et la dette cumulée.
4. Visualiser ce fichier grâce à la commande `plot-dettes.sh`. Le shell-script `plot-dettes.sh` appelle le programme `gnuplot` qui permet de créer des graphiques à partir de la ligne de commande.
5. Repérer le processus associé à l'affichage graphique. Comment terminer l'affichage sans saisir `q` dans la fenêtre `gnuplot` ?

Ex. 8 : Notes AB 20 min.

Le fichier `notes1.dat` contient les notes d'un groupe d'étudiants. Le point-virgule `";"` peut tenir lieu de séparateur entre colonnes (comme dans les fichiers au format CSV exportés par les tableurs).

notes1.dat

```
Nom Prenom Initiales;cc1;cc2;ex;oral
Legrand Jean JL;10;12;13;14
Dupont Yves YD;9;7;9;12
Durand Claire CD;8;12;11;10
Lefort Corine CL;15;14;14;12
```

1. Utiliser la commande `awk` pour permuter les notes d'oral et d'examen (`ex`). Comment restituer le séparateur « ; » entre les notes en sortie ?
2. Calculer la moyenne par épreuve sur l'ensemble du groupe d'étudiants et l'afficher.
3. B Calculer la moyenne des notes pour chaque étudiant et l'afficher.
4. Utiliser la commande `awk` pour permuter nom et prénom.
5. B Reprendre la permutation nom–prénom avec la commande `sed` et conclure.
6. B Reprendre la permutation nom–prénom avec `awk` puis `sed` sur le fichier `notes1+blancs.dat`. Expliquer.

TE 4
Le shell : les processus, les variables et leur portée

Références

— Polycopié UNIX : chapitres 10, 11, 12, ...

— Transparents UNIX : sections 16 à la fin

Créer un répertoire `te4` et y copier les fichiers fournis dans le répertoire `te4` du compte de l'UE.

Ex. 1 : Variable `PATH` A 20 min.

L'objectif est de tester le rôle de la variable d'environnement `PATH` sur le lancement des commandes système et utilisateur. Pour éviter de perturber la session avec ces tests, on lancera tout d'abord en arrière plan un terminal d'une couleur particulière, par exemple via la commande `xterm -bg Skyblue &` Toutes les commandes de cet exercice seront ensuite lancées à partir de ce terminal ou de l'un de ses fils.

1. Le shell mémorise les chemins d'accès aux commandes dans une table afin d'accélérer leur lancement : afficher cette table avec la commande interne `hash`. Lancer quelques commandes (`ls`, `date`, puis `ls`, ...) en surveillant l'évolution de la table avec `hash`.
   Puis réinitialiser cette table avec `hash -r` et vérifier qu'elle est vide (on pourra comparer avec la table de la session d'un autre terminal). Puis identifier quelques commandes avec la primitive `type` : `type date`, puis de même pour `xterm`, `xclock`, `cd`, `id`. Noter les résultats.
2. Afficher la liste des chemins de recherche des exécutables par `echo $PATH`. Comment afficher un chemin par ligne ?
   Puis réduire la liste des chemins de recherche des exécutables par `PATH="/bin"`
   Vérifier la nouvelle liste. Enfin lancer successivement les commandes `ls`, `xterm`, puis `/usr/bin/xterm`. Expliquer leur effet et en particulier les éventuels messages d'erreur.
3. Changer de répertoire de travail par `cd /usr/bin` Lancer `xterm`, puis `./xterm`. Expliquer.
4. Compléter la liste des chemins de recherche par `PATH="${PATH}:/usr/bin"`
   Lancer `xterm` et expliquer. Dans le dernier terminal lancé, afficher la liste des chemins de recherche par `echo $PATH`. Expliquer.
5. Ouvrir un terminal dit de « login », c'est-à-dire qui exécute les fichiers de démarrage (notamment `~/.bash_profile`)
   `xterm -ls` et remarquer l'affichage du mot du jour.
   Dans ce nouveau terminal, afficher la liste des chemins. Expliquer.
6. Copier le fichier source C `carre+invite.c` (utilisé en TE 2) dans votre répertoire `te4`. Quels droits avez-vous sur ce fichier ?
   Compiler ce fichier source via `gcc-mni-c89 carre+invite.c` Qui peut exécuter le fichier `a.out` produit ? Avez vous eu besoin d'utiliser `chmod` pour le rendre exécutable ? Essayer de le lancer avec `a.out`, puis avec `./a.out` ; expliquer.
   Ajouter le répertoire courant à la fin de la variable `PATH`. Peut-on lancer l'exécutable `a.out` avec seulement son nom sans préciser de chemin ? Même question après avoir renommé l'exécutable en `carre-c.x`
7. B Renommer provisoirement le fichier `a.out` en `ls`. Avec ce `PATH`, comment doit-on procéder pour le lancer ? Détruire votre fichier `ls`.

Fermer le terminal « bleu ciel » afin de retrouver une configuration plus saine.

Ex. 2 : Processus AB 10 min.

Se connecter sur le serveur d'applications pour effectuer cet exercice.
Lancer une horloge en arrière plan (`xclock &`), puis un terminal (`xterm &`). Dans ce terminal, lancer une autre horloge en arrière plan, puis un autre terminal `xterm` et enfin un terminal `uxterm`.
Exécuter la commande `ps -f -U` *votre_id* où *votre_id* est donné par la commande `id`[1]. Représenter la généalogie des processus indiqués (utiliser les PPID/PID **p**arent-/**p**rocess **id**entifier). Vérifier la hiérarchie des processus lancés avec l'option `--forest` sous LINUX. Comment afficher seulement les processus associés aux terminaux `xterm` ?
Interrompre les processus liés aux horloges avec la commande `kill`.

Ex. 3 : Calcul et contrôle de processus AB 20 min.

1. Copier les fichiers sources `infini-while.c` et `infini-while.f90` en langage C et fortran90 respectivement dans votre répertoire `te4`. Expliquer pourquoi ils produisent des boucles sans fin. Compiler un de ces fichiers pour obtenir un exécutable nommé `infini-while.x` :
   `gcc-mni-c89 infini-while.c -o infini-while-c.x`
   ou
   `gfortran03-mni infini-while.f90 -o infini-while-f.x`
   Lancer cet exécutable et attendre... Afficher les processus personnels actifs avec `jobs -l`. Surveiller la charge de la machine avec la commande `top`. On repère facilement dans `top` les processus d'un utilisateur en saisissant `u` suivi de son identifiant ou de son numéro d'utilisateur (UID) fourni par la commande `id -u`[2]. Il faudra interrompre le processus !
2. B Faire une copie de ces fichiers nommée `infini-while+.f90` (`infini-while+.c`) et changer la multiplication en addition dans la boucle ; compiler la copie en nommant l'exécutable `infini-while+f.x` (`infini-while+c.x`). Lancer successivement ces deux exécutables et attendre : expliquer leur comportement (penser à la représentation des entiers).

Ne pas oublier d'arrêter tous les processus `infini-while.x` lancés avant de se déconnecter.

Compléments du TE 4

Traiter les exercices 4 et 5 à la fois sur le serveur et sur la machine locale.

Ex. 4 : Premiers shell-scripts AB 10 min.

1. Un fichier de commande (ou shell-script) est un fichier texte contenant des commandes destinées au shell. Un shell-script a déjà été utilisé dans la question 3 de l'exercice 5 du TE 2.
   - Créer un fichier `premier.sh` contenant le texte ci-contre
   - Exécuter le script en saisissant : `bash premier.sh`
   - Lancer la commande `./premier.sh` Permet-elle l'exécution du script ? Ajuster les permissions du fichier pour autoriser son exécution via cette commande.

   premier.sh
   ```
   date
   hostname
   whoami
   cat /home/lefrere/M1/Config/motd
   ```

   - Lancer la commande `premier.sh` Permet-elle l'exécution du script ? Afficher la variable `PATH` par la commande `echo $PATH`.
   - Ajouter le répertoire courant dans la variable `PATH`, et effectuer de nouveau la commande `premier.sh` Conclure.
2. Les shell-scripts doivent être écrits en suivant une syntaxe particulière dépendant du type de shell utilisé. Pour connaitre le nom du shell interactif utilisé dans votre terminal, saisir `echo $SHELL`, et pour connaître le type de terminal utilisé, `echo $TERM`. Ces variables sont des variables d'environnement standard de même que `PATH`, `USER` et `HOME`.

1. Attention `ps -U` *votre_login* serait compris comme un `id` si le login est numérique, ce qui est le cas des étudiants.
2. L'argument `nom_de_login` de l'option `u` n'est pas utilisable avec les logins numériques mis en place à l'UPMC, qui seraient considérés comme des UID.

Afin d'assurer la portabilité du script, il est conseillé d'indiquer dans le fichier lui-même le shell [3] qui doit interpréter le script, quel que soit le shell père (depuis lequel le script est lancé) : `#!/bin/bash` placé sur la première ligne du shell-script. Sur une ligne, ce qui est écrit à la suite d'un `#` est considéré comme un commentaire. En particulier toute ligne commençant par un `#` est un commentaire... sauf si elle suit la syntaxe précédente (`#!`).
Créer le fichier `deux.sh` ci-contre.
Exécuter ce shell-script. Commenter.

deux.sh
```
#!/bin/bash
# Ceci est un commentaire
pwd
ls -l
# ceci est un autre commentaire
echo "--------------------------------"
cd
pwd
ls | grep 'mni' # encore un commentaire
echo "Fin du script"
```

3. Écrire un script `poids.sh` qui affiche sur la sortie standard : le nom de l'utilisateur, le chemin absolu de son répertoire d'accueil et l'espace occupé par l'ensemble des fichiers sous son répertoire d'accueil (utiliser la commande `du`).

Ex. 5 : Variables du shell **AB** **30 min.**

1. Créer un fichier `trois.sh` contenant

```
#!/bin/bash
# utilisation de la fonction echo
echo "La date du jour est: " date
echo "La date du jour est: " ; date
echo -n "La date du jour est: "; date
```

Exécuter ce fichier de commandes et commenter.

2. Créer un fichier `quatre.sh` contenant

début de quatre.sh
```
1 var1=un
2 var12=douze
3 echo ligne1: $var1
4 echo ligne2: $var1plusdutexte
5 echo ligne3: ${var1}
6 echo ligne4: ${var1plusdutexte}
7 echo ligne5: ${var1}plusdutexte
```

fin de quatre.sh
```
echo ligne6: ${var12}                                  8
echo ligne7: ${var1}2                                  9
echo ligne8: $var1et$var12                             10
echo ligne9: ${var1}et${var12}                         11
echo ligne10: ${var1} et ${var12}                      12
echo ligne11: ${var1}     et     ${var12}              13
echo ligne12: "${var1}     et     ${var12}"           14
```

Exécuter le shell-script et expliquer en détail les affichages.

3. Créer un fichier `cinq.sh` contenant

début de cinq.sh
```
1 #!/bin/bash
2 # La commande date +%X affiche
3 #  l'heure sous la forme H:M:S
4 var1=xxxxxxxx
5 date +%X
6 var1=date +%X
```

fin de cinq.sh
```
echo 1 ${var1}          7
var1="date +%X"         8
echo 2 ${var1}          9
var2=$(date +%X)        10
echo 3 ${var2}          11
var2=$(date +%D)        12
echo 4 ${var2}          13
```

Expliquer le comportement du shell-script ligne par ligne.

4. Écrire un script `espace.sh` pour obtenir un affichage du type :
`Mon nom de connexion est : numero_etudiant`
`Mon répertoire d'accueil est : /home/numero_etudiant`
`Espace utilisé: XXX  sur 500 Mo`

3. Le bash (Bourne Again Shell) est le shell par défaut de la plupart des distributions Linux, mais aussi celui du terminal de Mac OS X. Mais il existe d'autres types de shell tels que le Bourne Shell (sh), le Korn Shell (ksh), le C Shell (csh), le Tenex C Shell (tcsh) et le Z Shell (zsh).

5. La commande `read` implique une lecture au clavier c'est à dire une saisie par l'utilisateur dans le terminal. Elle permet de récupérer plusieurs variables saisies en une seule ligne. Dans le cas où l'utilisateur saisit plus de mots que demandés, la dernière variable contient les derniers mots de la ligne. Dans le cas où il n'en saisit pas suffisamment, les dernières variables sont vides.

   Créer un fichier `six.sh` contenant

```
#!/bin/bash
echo saisir trois mots
read varA varB varC
echo Affichage des valeurs saisies:
echo ${varA}
echo ${varB}
echo ${varC}
```

   Exécuter le script-shell en saisissant au clavier un, deux, trois et enfin quatre mots. Créer un fichier texte contenant une ligne de trois mots et utiliser une redirection afin d'éviter la saisie interactive.
6. Écrire un script-shell `compte_pixel.sh` qui compte le nombre de pixels à un niveau de gris donné dans un fichier d'image du type pgm en ascii et affiche ce nombre à l'écran. On utilisera le fichier `img.pgm` du TE 3. Dans un premier temps, le chemin du fichier et le niveau de gris recherché seront fixés dans le script. On vérifiera qu'il y a 16 pixels dont le niveau de gris est 200.
   Dans un second temps, ce script invitera l'utilisateur à saisir le chemin du fichier, puis le niveau de gris recherché. Si le script crée des fichiers temporaires, il doit également les effacer.

Ex. 6 : Les paramètres des scripts AB | 15 min.

1. Créer un fichier `huit.sh` contenant

```
#!/bin/bash
echo -n "la procédure $0 "
echo a été appelée avec $# paramètres
echo le premier paramètre est $1
echo le premier paramètre est $2
echo la liste des paramètres est $*
echo le numéro du processus lancé est $$
```

   Exécuter la commande :
   `huit.sh VAR 5 du texte`
   puis la commande :
   `huit.sh VAR 5 "du texte"`
   Expliquer.
2. Modifier le script-shell `compte_pixel.sh` pour qu'il utilise des paramètres fournis sur la ligne de commande dès le lancement du script plutôt qu'il attende leur saisie interactive au clavier.
3. Créer le shell-script `compile.sh` qui compile un programme C source dont le nom (sans le `.c`) est passé en paramètre. On suppose que le fichier source est dans le même répertoire que celui contenant le fichier `compile.sh`. Ce script placera l'exécutable dans le répertoire d'accueil, dans un fichier de même nom mais sans extension. Ce script devra afficher tout d'abord son nom, le paramètre passé, puis la taille du fichier source et le nombre de lignes qu'il contient. Il lancera ensuite la compilation (via `gcc`) et placera l'exécutable comme demandé. Il affichera enfin la taille du fichier exécutable. Utiliser le fichier `carre+invite.c` (*cf.* TE 2) pour tester le script.

Ex. 7 : Combinaisons de commandes et commande `exit` AB | 10 min.

1. Exécuter dans votre terminal les commandes ci-contre :
   Rappeler ce que représente `$?`

```
ls /inexistant
echo $?
cat /home/lefrere/M1/Config/motd
echo $?
```

2. Exécuter ensuite dans votre terminal les commandes :
   Parmi les 4 commandes précédentes pourquoi certaines affichent `Yes` alors que d'autres non ?

```
ls /inexistant && echo 'Yes'
ls /inexistant || echo 'Yes'
cat /home/lefrere/M1/Config/motd && echo 'Yes'
cat /home/lefrere/M1/Config/motd || echo 'Yes'
```

3. En utilisant les combinaisons de commandes `&&` et `||` ainsi que la commande `exit`, modifier `compile.sh` afin que le déplacement dans le répertoire d'accueil ainsi que l'affichage du fichier exécutable ne se fasse que si la compilation a réussi.