Initiation à Linux

Guide de survie

```
$ cd ~
$ touch indice.txt
$ echo "Khrouchtchev n'Aimait
pas beaucoup les Américains
Mais il détestait Encore plus
Les Orties de sa Tante Tatiana"
> indice.txt
$ tr -sc 'A-Z' '.' < indice.txt
```

Contact :

pierre-antoine.jean

Le système d'exploitation Linux

Roparzh

"*Un système d'exploitation* ? Qu'est-ce que vous voulez-vous insinuyer Sire ?"

Un système d'exploitation (*OS*) est un ensemble de programmes qui coordonnent le fonctionnement des différents composants matériels et logiciels d'un système informatique

Le système d'exploitation gère:

- L'allocation de la **mémoire**: partage de la mémoire entre les programmes
- Les **périphériques**: écran, imprimante, disque dur, réseau. L'*OS* s'assure que les programmes puissent les utiliser de façon standard
- Le **processeur**: utilisation partagée entre tous les programmes (planification des processus)
- Les **utilisateurs**: gestion des droits d'accès aux fichiers et aux matériels

Le système d'exploitation Linux

- Créé en 1991 par Linus Torvalds
- Système d'exploitation *open-source* dérivé du système d'exploitation Unix
- Existe de nombreuses distributions Linux

Une distribution Linux est une suite de logiciels assemblés autour du noyau Linux pour former un système d'exploitation pleinement opérationnel

- Chaque distribution a sa propre philosophie et répond à des besoins spécifiques

Ubuntu : Commercialisé et maintenu par la société Canonical, cette distribution est dérivée d'une autre distribution nommée Debian. Elle propose tous les 6 mois une version stable (maintenue 9 mois) et tous les 2 ans une version LTS (maintenue plusieurs années). Parfait pour découvrir Linux

ArchLinux : Distribution communautaire sans version, elle est en mise à jour permanente grâce à une communauté active de développeurs. Convient à un public davantage expérimenté

Plan

Le système de fichiers

Les commandes importantes

Les scripts

Les références

Plan

Le système de fichiers

Les commandes importantes

Les scripts

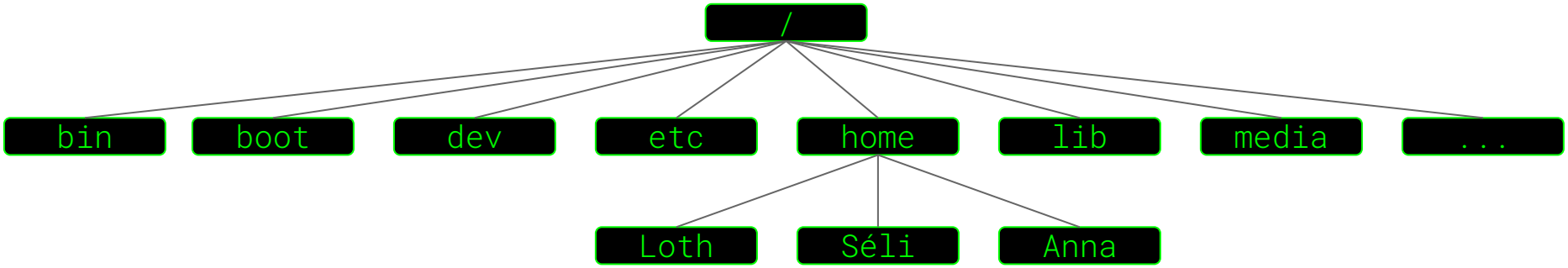
Les références

Le système de fichiers - Une arborescence

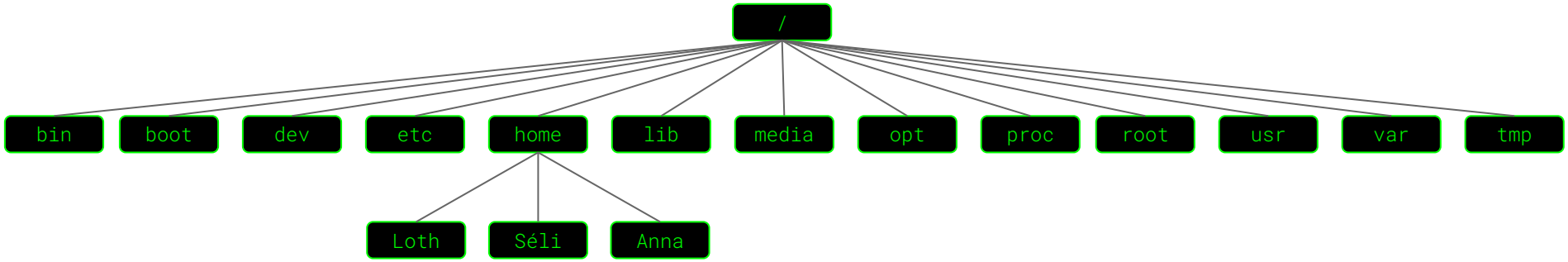
Le système de fichiers correspond à une arborescence que l'on parcourt de la racine (`root` ou "/") vers les feuilles

Perceval

"Comment ? Y'a des racines ?! Mais le vieux, y va pas me filer un panneau, ça va faire con."



Le système de fichiers - Une arborescence



`/bin` : Exécutables essentiels pour le système et utilisables par l'utilisateur
`/boot` : Fichiers permettant au système d'exploitation de démarrer
`/dev` : Points d'entrée des périphériques
`/etc` : Fichiers de configuration du système et du réseau
`/home` : Répertoires personnels des utilisateurs
`/lib` : Principales bibliothèques partagées (essentielles au système lors du démarrage)
`/media` : Points de montage des partitions temporaires (*e.g.* clés usb)
`/opt` : *Packages* d'applications supplémentaires (installation des logiciels commerciaux)
`/proc` : Fichiers contenant des informations sur l'état du système et des processus en cours
`/root` : Répertoire de l'administrateur système
`/usr` : Données que les utilisateurs peuvent se partager (documentation, jeux, *etc.*)
`/var` : Données fréquemment réécrites (*e.g.* logs)
`/tmp` : Fichiers temporaires

Le système de fichiers - Les chemins d'accès

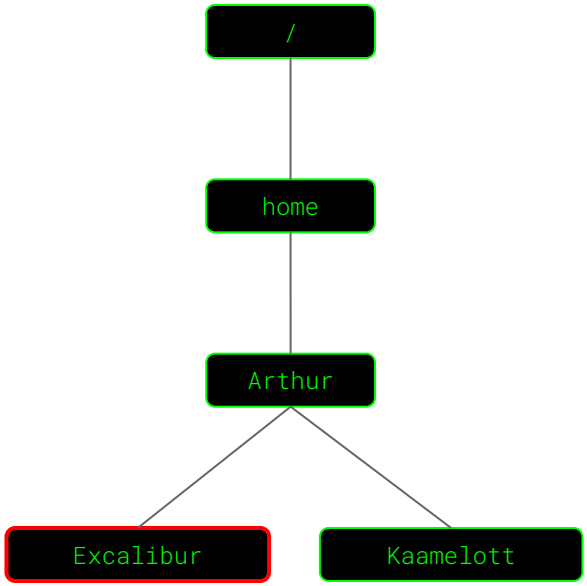
- Notion de **chemin d'accès** pour identifier un fichier: suite de noms étiquetant les fichiers le long de l'arborescence
- Référence absolue correspond au chemin d'accès depuis la racine "/"
  e.g. `/home/Perceval`
- Référence relative correspond au chemin d'accès depuis le répertoire de travail
  + Le répertoire courant est symbolisé " **.** "
  + Le répertoire parent est symbolisé " **..** "
  + Le répertoire personnel est symbolisé par " **~** " (`/home/Perceval`)

  e.g. `../../home/Documents/Projets`
  `~/Documents/Projets`
- Les fichiers cachés ont un " . " devant leur nom de fichier (*e.g.* `/home/Perceval/.bashrc`)

Perceval

"Bon, alors maintenant ; la technique du rebrousse-chemin."

Le système de fichiers - Les chemins d'accès



L'utilisateur se situe au sein du dossier `Excalibur`
Le chemin absolu de ce dossier est :

`/home/Arthur/Excalibur`
ou
`~/Excalibur`

Le dossier parent de ce dossier est :

`../` → `Arthur`

Le chemin relatif vers le dossier `Kaamelott` est :

`../Kaamelott`

Plan

Le système de fichiers

Les commandes importantes

Les scripts

Les références

Dialoguer avec le système - Le *shell*

L'utilisateur dialogue avec le coeur du système (noyau) par l'intermédiaire du *shell* (ou l'interpréteur de commandes / terminal)

Roi Burgonde

"*Shell* ? Qu'est ce à dire que ceci ?"

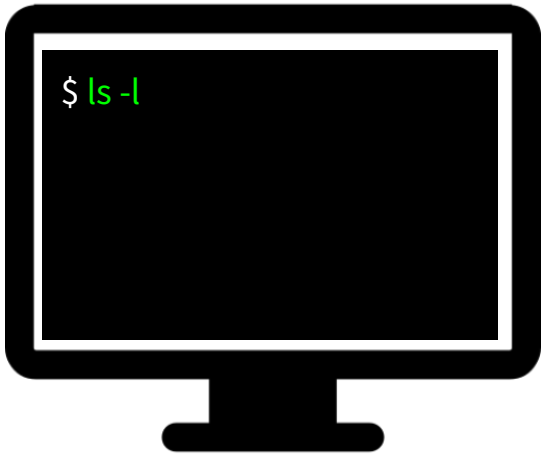
Un utilisateur saisit sa commande (le symbole "$" symbolise l'invite du *shell*)

Analyse et exécution de la commande correspondante

Résultat de la commande `ls` pour lister le contenu d'un répertoire

Utilisation du *shell*

- Le *shell* correspond à une fenêtre présentant un *prompt* (invite de commande)



Utilisateur normal

Super-utilisateur

- Les commandes sont saisies à la suite dans le *prompt*
- Les touches ↑ et ↓ permettent de faire défiler la liste des commandes précédemment tapées (visibles dans `~/.bash_history`)

Les commandes - Introduction

- Une commande est un **programme**

  nom_commande [-liste_options] [liste_arguments]

- Sensibilité à la **casse**

  `cd ~` ≠ `CD ~`

- Une commande crée un processus et tous les processus possèdent **3 flux standards**: un d'entrée - *stdin* de valeur 0 - (typiquement le clavier) et deux de sortie (la sortie standard - *stdout* de valeur 1 - et l'erreur - *stderr* de valeur 2 - étant généralement l'affichage)

Les commandes - Les redirections des flux E/S

> Redirection de la sortie standard

`$ command > output.txt` : Redirige la sortie standard de `command` dans le fichier `output.txt` (écrase le contenu du fichier)

2> Redirection de la sortie standard erreur

`$ command 2> errors.txt` : Redirige les messages d'erreurs de `command` dans le fichier `errors.txt` (écrase le contenu du fichier)

&> Redirection à la fois de la sortie standard et de la sortie standard erreur

`$ command &> output.txt` : Redirige la sortie standard de `command` ainsi que ses messages d'erreurs dans le fichier `output.txt`

`$ find / -name "*.conf" > output.txt 2> errors.txt` : Redirection de `stdout` dans le fichier `output.txt` et de `stderr` dans le fichier `errors.txt`

>> Redirection de la sortie standard avec concaténation (ajoute à la suite du fichier)

2>> Redirection de la sortie standard erreur avec concaténation (ajoute à la suite du fichier)

Les commandes - Les redirections des flux E/S

< Redirection de l'entrée standard

```
$ cat < tp1.txt
```
: `cat` affiche le contenu du fichier `tp1.txt`

| Redirection de la sortie standard d'une première commande vers l'entrée standard d'une seconde commande

```
$ cat tp1.txt | more
```

Perceval

"Faut arrêter ces conneries de *stdout* et de *stderr* ! Une fois pour toutes, *stdout*, suivant comment on est tourné, ça change tout !"

Les commandes - Introduction

Besoin d'un renseignement sur une commande ? Affichez la documentation d'une commande (description, visualisation des options, *etc.*) avec :

```
man <nom_commande>
```

Perceval

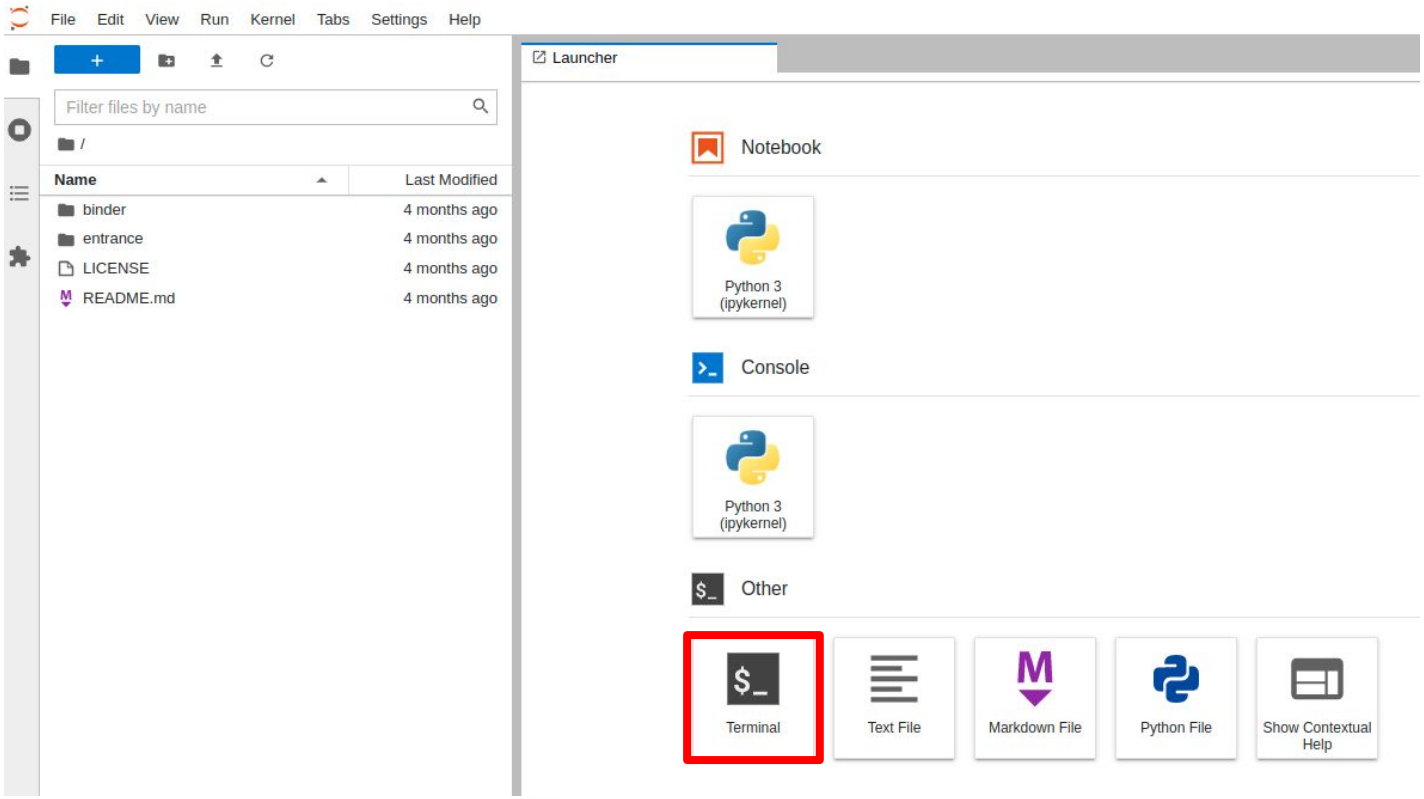
"Moi, la *commande* `man`, ça m'aide. Je visualise la *commande* dans le *shell*, j'ai l'impression de faire partie d'un tout, moi, la *commande*, le *shell*, le *noyau*, la *couche matérielle*, c'est entier, vous comprenez ?"

Besoin des droits administrateur ? Faites précéder votre commande par *sudo*:

```
sudo ls -la
```

Les commandes - Serious Game, BashCrawler

Rendez-vous → [https://gitlab.com/slackermedia/bashcrawl]. Dans la section "*Try it online with mybinder*", cliquer sur le lien pour accéder au JupyterLab



Ouvrir un nouveau terminal et suivre les instructions !

Les commandes - Parcourir l'arborescence des répertoires

- Où suis-je dans l'arborescence et qui suis-je ?

```
$ pwd
/home/Perceval
$ who
Perceval
```

- Changer de répertoire

```
$ cd Documents
$ pwd
/home/Perceval/Documents
$ cd
$ pwd
/home/Perceval
```

- Créer un répertoire

```
$ mkdir Projets
$ cd Projets
```

- Créer un fichier

```
$ touch tp1.txt
```

Les commandes - Rechercher dans l'arborescence

- La commande `find` permet d'effectuer une recherche sur l'arborescence
  `$ find . -name tp1.txt` : Rechercher dans le répertoire courant et ses sous-répertoires un fichier nommé `tp1.txt`
  `$ find ~/Documents/Projets/ -type d` : Rechercher tous les sous-répertoires du répertoire `~/Documents/Projets/`
  `$ find . -exec ls -l '{}' \;` : Exécute la commande `ls -l` sur tous les fichiers trouvés (l'option `-exec cmd '{}'` doit se terminer avec `\;`)

- Retrouver un fichier exécutable avec la commande `which`

```
$ which ls
/bin/ls
```

Kadoc

"Elle est où la poulette ?"

Les commandes - Manipulation de fichiers

- Afficher la liste des fichiers
  `$ ls` : Afficher la liste des fichiers et sous-répertoires du répertoire courant
  `$ ls ~/Documents` : Afficher la liste des fichiers et sous-répertoires du répertoire `/home/Perceval/Documents`
  `$ ls -l` : Afficher la liste détaillée (droits, propriétaire, groupe, taille, *etc.*)
  `$ ls -a` : Afficher également les fichiers cachés (*e.g.* `/home/Perceval/.bashrc`)
- Copier un fichier
  `$ cp tp1.txt ../` : Copier le fichier `tp1.txt` dans le répertoire parent
  `$ cp -R Documents /tmp/` : Copier toute l'arborescence du répertoire `Documents` dans le répertoire `/tmp/`
- Déplacer et renommer un fichier
  `$ mv tp1.txt tp_1.txt` : Renomme `tp1.txt` en `tp_1.txt`
  `$ mv tp1.txt ../` : Déplace `tp1.txt` vers le répertoire parent

Les commandes - Manipulation de fichiers

- Détruire un fichier
  `$ rm tp1.txt` : Détruit le fichier `tp1.txt`
  `$ rm ~/Documents/*.png` : Supprime tous les fichiers avec l'extension `.png` du répertoire `~/Documents`
  `$ rm -r ~/Documents/Projets` : Détruit récursivement le répertoire `Projets`

Léodagan

"`rm -rf /` . Tout cramer et reprendre à zéro. Si j'étais Roi, c'est ce que je ferais avec la Bretagne."

- Créer un lien symbolique à partir du **chemin absolu** d'un autre fichier où toute opération sur ce fichier "symbolique" (lecture, écriture, *etc.*) s'effectue sur le fichier référencé (équivalent aux raccourcis sous Windows)
  `$ ln -s ~/Documents/Projets/tp1.txt ../tp1.txt`

Les commandes - Manipulation de fichiers

- Examiner un fichier sans le modifier
  `$ cat tp1.txt` : Affiche le contenu du fichier `tp1.txt` (permet également la concaténation de fichiers)
  `$ more tp1.txt` : Permet de parcourir le fichier `tp1.txt` ligne par ligne ou page par page
  `$ head -10 tp1.txt` : Affiche les 10 **premières** lignes du fichier `tp1.txt`
  `$ tail -10 tp1.txt` : Affiche les 10 **dernières** lignes du fichier `tp1.txt`
- Éditer et modifier un fichier avec un éditeur

```
$ (x)emacs tp1.txt
$ vi(m) tp1.txt
$ nano tp1.txt
$ pico tp1.txt
```

- La commande `echo` affiche sur la sortie standard les messages passés en paramètre (après leur interprétation par le shell)
  `$ echo -e "Bonjour:\n$USER"` : L'option -e permet d'interpréter les caractères échappés (\n, \t, *etc.*)

Les commandes - Manipuler le contenu d'un fichier

- Compter dans un fichier avec la commande `wc`
  - `$ wc -l tp1.txt` : Compter le nombre de lignes
  - `$ wc -w tp1.txt` : Compter le nombre de mots
  - `$ wc -m tp1.txt` : Compter le nombre de caractères
  - `$ wc -c tp1.txt` : Compter le nombre d'octets

- Trier les lignes d'un fichier avec la commande `sort`
  - `$ sort tp1.txt` : Trie alphanumérique croissant
  - `$ sort -r tp1.txt` : Trie alphanumérique décroissant

`Personnages.txt`

```
Yvain|Gauvain
Perceval|Karadoc
Merlin|Elias
```

```
$ sort Personnages.txt
```

```
Merlin|Elias
Perceval|Karadoc
Yvain|Gauvain
```

`Personnages.txt`

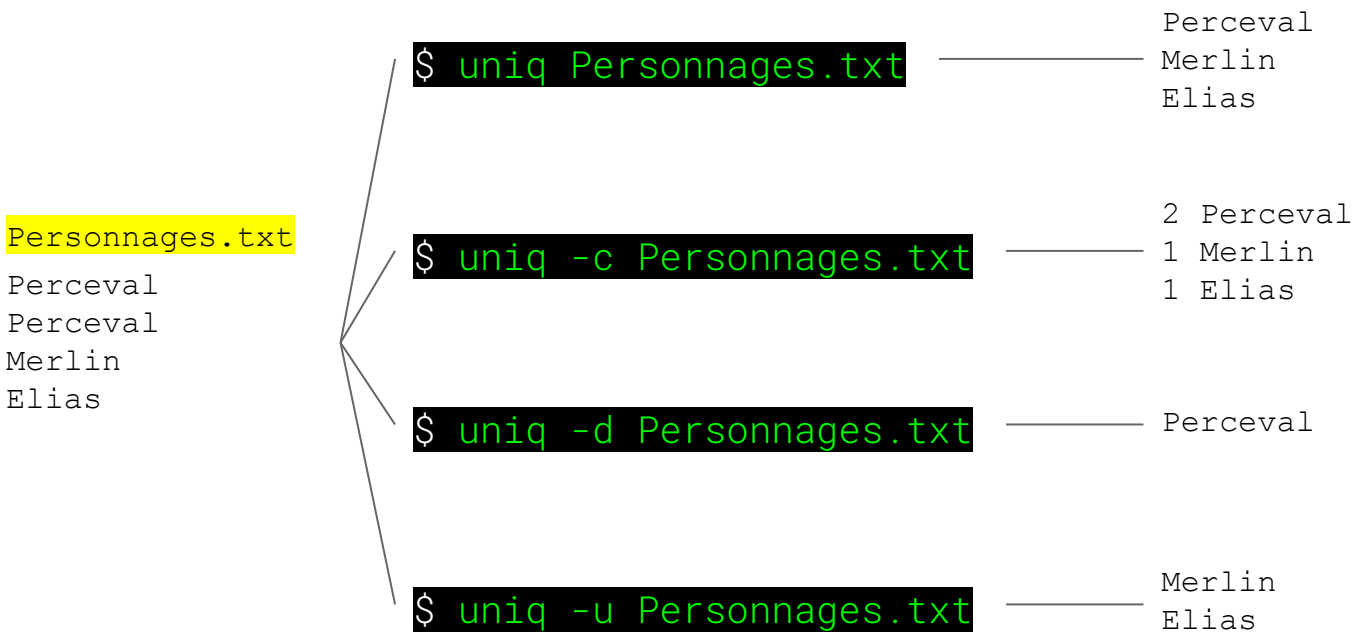
```
Yvain|Gauvain
Perceval|Karadoc
Merlin|Elias
```

```
$ sort -t'|' +1.1 Personnages.txt
```

```
Perceval|Karadoc
Yvain|Gauvain
Merlin|Elias
```

Les commandes - Manipuler le contenu d'un fichier

Supprimer les doublons **qui se suivent** dans les fichiers texte avec la commande `uniq`



Les commandes - Manipuler le contenu d'un fichier

Convertir une chaîne de caractères avec la commande `tr`

- Convertir tous les caractères "*a*" en caractère "b" dans le fichier

```
$ echo "abcd" | tr 'a' 'b'
bbcd
```

- Convertir tous les caractères qui ne sont pas spécifiés dans la première chaîne selon les caractères de la seconde

```
$ echo "abcd" | tr -c 'a' 'b'
abbbb
```
(conversion du caractère `\n`)

- Effacer un caractère spécifié

```
$ echo "abcd" | tr -d 'a'
bcd
```

- Réduire à une seule unité un caractère spécifié qui se répète **plusieurs fois de suite**

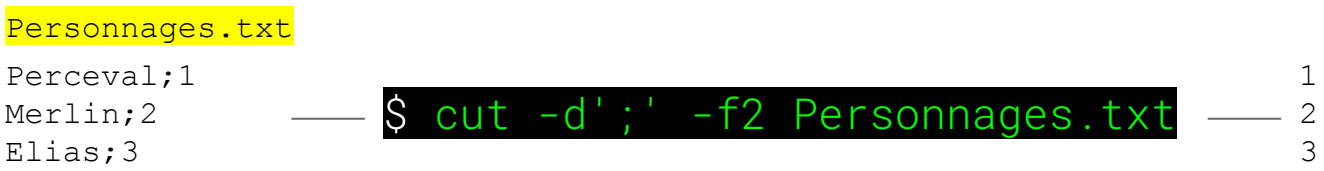
```
echo "les pattes de canaaaaaard" | tr -s 'a'
les pattes de canard
```

Kadoc

"Les pattes de canaaaaaaaaaaaaaaaaaaaaaaaaaaaard !"

Les commandes - Manipuler le contenu d'un fichier

- Afficher des zones spécifiques d'un fichier avec la commande `cut`
  `$ cut -c1-5 tp1.txt` : Affiche les colonnes de 1 à 5
  `$ cut -d';' -f2 tp1.txt` : Affiche la colonne 2 en utilisant le séparateur " ; "



- La commande `sed` est un éditeur non interactif (aucun visuel) elle permet d'appliquer différents traitements sur un fichier puis d'en afficher le résultat (sans modification du fichier de départ) sur la sortie standard
  `$ sed "s/[Pp]rovencal/Perceval/g" tp1.txt` : Permet de substituer toutes les occurrences de *Provencal*/*provencal* par *Perceval*
  `$ sed -e "2,4d" tp1.txt` : Affiche toutes les lignes du fichier sauf les lignes 2, 3 et 4

Les commandes - Manipuler le contenu d'un fichier

A l'aide des commandes précédentes et du système de redirection des flux, répondre aux questions suivantes :

Vous avez un fichier Personnages.txt avec le contenu suivant :

```
Lancelot;Du Lac
Bohort;De Gaunes
Lionel;De Gaunes
Aconia;Minor
```

1) Afficher dans l'ordre alphabétique uniquement les noms de famille des personnages.
2) Afficher dans l'ordre alphabétique uniquement les noms de famille des personnages en supprimant les doublons.
3) Afficher dans l'ordre alphabétique uniquement les noms de famille des personnages en supprimant les doublons et en convertissant les espaces par des *underscores* (_).

Les commandes - Manipuler le contenu d'un fichier

- Rechercher une chaîne de caractères dans un fichier avec la commande `grep`

  `$ grep -i "couhillère" tp1.txt` : Recherche l'expression "couhillère" dans le fichier `tp1.txt` et ignore la casse (-i)

Roi Burgonde

"ARTHOUR ! … Couhillère !"

`$ grep -v "couhillère" tp1.txt` : Affiche les lignes ne contenant pas la chaîne

`$ grep -c "couhillère" tp1.txt` : Compte le nombre de lignes contenant la chaîne

`$ grep -n "couhillère" tp1.txt` : Chaque ligne contenant la chaîne est numérotée

`$ grep -P "il+a"` : Expression régulière étendue (*Perl*)

`$ grep -Po "\..*$"` : L'option '*-o*' permet d'afficher uniquement le motif correspondant contenu au sein d'une phrase donnée

Les commandes - Manipuler le contenu d'un fichier

Les expressions régulières

Les caractères de début et de fin de chaîne

```
^Bonjour
revoir$
```

Le caractère OU

```
Bonjour|revoir
^Bonjour|revoir$
```

Les ensembles de caractères

```
mots|mats|mits
m[oai]ts
m[^oai]ts
[a-z] équivalent à [abcdefghijklmonpqrstuvwxyz]
[A-Z] équivalent à [ABCDEFGHIJKLMNOPQRSTUVWXYZ]
[0-9] équivalent à [0123456789]
[a-z0-9] équivalent à [abcdefghijklmonpqrstuvwxyz0123456789]
 . équivalent à Absolument n'importe quel caractère
\w équivalent à [a-zA-Z0-9_]
\d équivalent à [0-9]
```

Les commandes - Manipuler le contenu d'un fichier

Les expressions régulières

Les quantificateurs

```
{min, max} ou {min,} ou {,max} ou {nombre}
* équivalent à 0 ou plusieurs répétitions soit à {0,}
+ équivalent à 1 ou plusieurs répétitions soit à {1,}
? équivalent à 0 ou 1 répétition soit à {,1}
```

L'échappement

```
johndoe.[a-z]{2,3} → johndoe\.[a-z]{2,3}
```

Les commandes - Manipuler le contenu d'un fichier

Les expressions régulières

" **couhillère** " ou " **couillère** " ?

```
$ grep "cou.*illère" tp1.txt
couhillère
couillère
```

`$ grep "."` : *Match* tous les caractères (sauf les sauts de ligne)
`$ grep "a"` : *Match* la lettre "a"
`$ grep "il*a"` : *Match* les chaînes "ia", "ila", illa", illla", "illlla", *etc.* (0 à n)
`$ grep -P "il+a"` : *Match* les chaînes "ila", illa", illla", "illlla", *etc.* (1 à n)
`$ grep "[a-z]"` : *Match* toutes les lettres minuscules (non accentuées)
`$ grep "[A-Z]"` : *Match* toutes les lettres majuscules (non accentuées)
`$ grep "[0-9]"` : *Match* tous les numéros
`$ grep "[aeiouyAEIOUY]"` : *Match* toutes les voyelles
`$ grep "[^aeiouyAEIOUY]"` : *Match* tous les caractères sauf les voyelles
`$ grep "^A"` : *Match* toutes les phrases débutant avec un "A"
`$ grep "A$"` : *Match* toutes les phrases terminant avec un "A"
`$ grep -P "l{1,2}"` : *Match* l'apparition de "l" et de "ll"
`$ grep -P "(ll)[a-z]\1"` : *Match* les termes contenant la chaîne "*ll…ll*" ou "…" est un caractère alphabétique en minuscule

Les commandes - Manipuler le contenu d'un fichier

Rendez-vous → [https://regex101.com/] et copier/coller le texte présent dans le fichier `regex.txt` du répertoire GitHub (à partir de l'affichage *Raw*).

Répondez aux questions suivantes :

1) Surligner les dialogues du Père Blaise.
2) Surligner les dialogues du Père Blaise et de Léodagan.
3) Surligner toutes les phrases qui ne commencent pas par un "P".
4) Surligner les dialogues du Père Blaise qui se terminent par un "?".
5) Surligner les chiffres au sein des dialogues.
6) Surligner les numéros de téléphone.
7) Surligner les adresses mail.

Les commandes - Le système de droits

Mise en situation

```
Perceval@machine $ cd /home/Arthur
cd: /home/Arthur/: Permission denied
Perceval@machine $ ls /home/Arthur
ls: /home/Arthur/: Permission denied
Perceval@machine $ ls -l /home
drwxr-xr-x 29 Perceval groupe1 4096 juil. 25 09:00 Perceval/
drwx------ 29 Arthur groupe2 4096 mai. 21 18:30 Arthur/
```

Nature du fichier
- “-” fichier normal
- “d” répertoire
- “l” lien symbolique

Perceval n’a pas les droits nécessaires pour accéder au répertoire de Arthur

Perceval

“C’est pas faux !”

Les commandes - Le système de droits

- Mise en situation

```
Perceval@machine $ ls -l /home
drwxr-xr-x 29 Perceval groupe1 4096 juil. 25 09:00 Perceval/
drwx------ 29 Arthur groupe2 4096 mai. 21 18:30 Arthur/
```

- Il existe 3 types de droits applicables à 3 classes d'utilisateurs
  + "*r*": droit de lecture dans un fichier
  + "*w*": droit d'écriture dans un fichier
  + "*x*": droit d'exécution
  + "-": absence d'un droit

- Répétés 3 fois pour indiquer le(s) personne(s) impliquée(s)

→ Le propriétaire (*Perceval*)

```
drwxr-xr-x
```

→ Les utilisateurs appartenant au même groupe que le propriétaire (*groupe1*)

```
drwxr-xr-x
```

→ Le reste des utilisateurs

```
drwxr-xr-x
```

Les commandes - Le système de droits

- La commande `cd /home/Arthur` nécessite le droit d'exécution (*x*) sur le répertoire
- La commande `ls /home/Arthur` nécessite le droit de lecture
- Uniquement le propriétaire et le `root` peuvent changer les droits d'un fichier/répertoire
- La command `chmod` permet de changer les droits d'un fichier/répertoire

`$ chmod o+x /home/Arthur` : Cette commande exécutée par Arthur accorde le droit d'exécution à tous les utilisateurs sur son répertoire personnel

`$ chmod o+x` :

`u` change les droits du propriétaire
`g` change les droits du groupe du propriétaire
`o` change les droits de tous les autres

`$ chmod o+x` :

`+` ajouter un droit
`-` enlever un droit

`$ chmod o+x` : Symbole des différents droits `r`, `w` ou `x`

Les commandes - Gestion des processus

- Un **processus** est une instance d'un programme en train de s'exécuter, il est identifié par un `PID` (*process ID*)
- La commande `ps` permet d'afficher des informations à propos d'une sélection de processus actifs

  `$ ps -e` : Afficher tous les processus avec leur `PID`

  `$ ps -u` : Afficher la liste détaillée (`PID`, occupation mémoire, consommation CPU, *etc.*) de tous les processus dont l'utilisateur est propriétaire
- La commande `top` permet de lister en temps réel les processus et les ressources utilisées
- La commande kill permet de tuer un processus en fonction de son `PID`

  ```
  $ kill -9 8384
  ```
- Une commande en cours d'exécution peut être arrêtée avec la manipulation `Ctrl-C` (en mode interactif)
- Lancer une commande en tâche de fond avec le symbole "**&**" (pour continuer à utiliser le *shell*)

Les commandes - Communiquer par le réseau

- La commande `ssh` (*Secure SHell*) est un programme et un protocole de communication sécurisée, elle permet de se connecter à un serveur/système distant afin d'obtenir un *shell*

```
$ ssh <nom_utilisateur>@<adresse_ip>
```

  La connection nécessite l'installation d'un serveur `ssh` (serveur qui une fois installé se lance généralement au démarrage de la machine dans le cas contraire `$ systemctl start sshd`)

- La commande `scp` permet de réaliser des transferts sécurisés via une connexion `ssh`

```
$ scp <nom_utilisateur>@<adresse_ip>:<chemin_fichier_serveur> <chemin_destination>
```

- La commande `wget` est une commande permettant de télécharger des fichiers via les protocoles HTTP, HTTPS et FTP

```
$ wget ftp://ftp.linux-france.org/pub/article/debutant/ini-unix.pdf
```

Les commandes - L'archivage et la compression

- L'**archivage** ne consiste pas à réduire l'espace disque d'un fichier, mais de rassembler plusieurs fichiers en un seul

  `$ tar -cf documents.tar ~/Documents/` : Archiver le répertoire `~/Documents` et ses sous-répertoires

  `$ tar -xf documents.tar` : Désarchiver `documents.tar`

  `$ tar -zcf documents.tgz ~/Documents/` : Archiver et compresser (gzip)
- La **compression** d'un fichier consiste à diminuer l'espace disque qu'il occupe (le principe de base est d'éviter les redondances en transformant une suite de `bits` A en une suite de `bits` B plus courte pouvant restituer les mêmes informations)
  + La commande gzip est un outil de compression de fichier unique

    `$ gzip tp1.txt` : Compresser un fichier

    `$ gunzip tp1.gz` : Décompresser un fichier
  + La commande zip est un programme d'archivage et de compression

    `$ zip documents.zip ~/Documents/` : Archive et compresse

    `$ unzip documents.zip` : Désarchive et décompresse

Les commandes - Variables d'environnement

- Une variable d'environnement est une variable accessible par tous les processus fils du *shell* courant
- Pour créer une variable d'environnement, on exporte la valeur d'une variable avec la commande `export`

  `$ export -p` : Liste tous les noms qui ont été exportés dans le *shell* actuel

  `$LANG` : détermine la langue que les logiciels utilisent pour communiquer avec l'utilisateur

  `$PATH` : Le système cherche les commandes tapées dans les dossiers spécifiés par la variable `$PATH`, dans l'ordre où ils sont indiqués

  `$HOME` : Emplacement du répertoire personnel de l'utilisateur actuellement connecté

  `$ export EDITOR=nano` : Exécute l'éditeur `nano` jusqu'à ce que la session se termine
- La persistance des `export` est effective lorsqu'ils sont indiqués dans le fichier `~/.bashrc` ou `~/.profile`

Les commandes - Routines

- Les `alias` sont des substitutions abrégées de commandes répétitives et/ou longues à taper dans la **console**
- La persistance des `alias` est effective lorsqu'ils sont indiqués dans le fichier `~/.bashrc` ou `~/.bash_aliases`

```
$ alias curdir='pwd | tr "/" "\n" | tail -1'
```

Perceval

"Dans le Languedoc, ils m'appellent Provençal. Mais c'est moi qui m'suis gouré en disant mon nom. Sinon en Bretagne, c'est le Gros Faisan au sud et au nord, c'est juste Ducon."

- Le Cron est un programme permettant de planifier automatiquement des tâches régulières

`$ crontab -l` : Lister la liste des tâches de l'utilisateur actuel

`$ crontab -e` : Éditer la liste des tâches de l'utilisateur actuel

```
$ 59 23 * * * echo "cron" >> ~/Documents/tp1.txt &> /dev/null
```

: Tous les jours à 23h59 ajouter le mot "cron" à la suite du fichier `tp1.txt` sans notification

Minute, Heure, Jours dans le mois, Mois, Jour de la semaine.

Les commandes - Le gestionnaire de paquets

- La distribution `debian` (et ses dérivés dont `Ubuntu`) exploite un gestionnaire de paquets nommé `apt` (*Advanced Packaging Tool*)

- Un gestionnaire de paquets est un système qui permet d'installer des logiciels, de les maintenir à jour et de les désinstaller

  `$ sudo apt update` : Rechercher les mises à jour des paquets installés

  `$ sudo apt search <mot_cle>` : Rechercher les paquets disponibles en fonction d'une expression

  `$ sudo apt install tree` : Installer un paquet à partir de son nom

  `$ sudo apt remove tree` : Supprimer un paquet installé sur l'ordinateur

Perceval

"On va pas installer notre carré germinal à la taverne !"

Les commandes - Autres

- La commande `rev` inverse une chaîne de caractères

```
$ echo "sorg a ne no eris" | rev
```

- La commande `sleep` permet de réaliser une pause pendant *x* secondes

```
$ sleep 10
```

- La commande `seq` génère une suite de nombres

```
$ seq 0 2
```

- La commande `shutdown -h now` éteint l'ordinateur

```
$ shutdown -h now
```

- La commande `reboot` redémarre l'ordinateur

```
$ reboot
```

|| Exécuter la commande suivante si la première a échoué

```
$ ls tp2 || echo "Pas de fichier tp2."
ls: tp2: No such file or directory
Pas de répertoire tp2.
```

&& Exécute la commande suivante uniquement si la première a réussi

```
$ ls tp1 && echo "J'ai un répertoire tp1."
(empty)
J'ai un répertoire tp1.
```

Plan

Le système de fichiers

Les commandes importantes

Les scripts

Les références

Les scripts

Yvain

“Est-ce qu'on peut s'en servir pour donner de l'élan à un pigeon ?”

Les scripts

Exécuter un ensemble d'instructions à partir d'un script *bash*

```
$ echo $SHELL
/bin/bash
```

graal.sh

```
#!/bin/bash

length=`ls ~/Documents|wc -l`

if [ $length -gt 2 ]
then
     echo "Le Graal!"
else
     echo "C'est de la merde."
fi

liste_fichiers=$(ls)
for fichier in $liste_fichiers
do
     echo "Fichier: $fichier"
done
```

Shebang (contraction entre *shell* et *bang*) symbolisé par "#!" permet de préciser l'interpréteur à utiliser

Les scripts

Exécuter un ensemble d'instructions à partir d'un script *bash*

```
$ echo $SHELL
/bin/bash
```

graal.sh

```
#!/bin/bash

length=`ls ~/Documents|wc -l`

if [ $length -gt 2 ]
then
     echo "Le Graal!"
else
     echo "C'est de la merde."
fi

liste_fichiers=$(ls)
for fichier in $liste_fichiers
do
     echo "Fichier: $fichier"
done
```

Déclaration de la variable `length` à partir de l'exécution d'une commande spécifiée avec les *back quotes* ``

Les scripts

Exécuter un ensemble d'instructions à partir d'un script *bash*

```
$ echo $SHELL
/bin/bash
```

graal.sh

```
#!/bin/bash

length=`ls ~/Documents|wc -l`

if [ $length -gt 2 ]
then
     echo "Le Graal!"
else
     echo "C'est de la merde."
fi

liste_fichiers=$(ls)
for fichier in $liste_fichiers
do
     echo "Fichier: $fichier"
done
```

Condition sur la variable `$length` en utilisant l'opérateur “>” avec `-gt` (*greater than*)

Les scripts

Exécuter un ensemble d'instructions à partir d'un script *bash*

```
$ echo $SHELL
/bin/bash
```

graal.sh

```
#!/bin/bash

length=`ls ~/Documents|wc -l`

if [ $length -gt 2 ]
then
     echo "Le Graal!"
else
     echo "C'est de la merde."
fi

liste_fichiers=$(ls)
for fichier in $liste_fichiers
do
     echo "Fichier: $fichier"
done
```

Déclaration de la variable `liste_fichiers` à partir de l'exécution d'une commande spécifiée avec `$( )`

Les scripts

Exécuter un ensemble d'instructions à partir d'un script *bash*

```
$ echo $SHELL
/bin/bash
```

graal.sh

```
#!/bin/bash

length=`ls ~/Documents|wc -l`

if [ $length -gt 2 ]
then
     echo "Le Graal!"
else
     echo "C'est de la merde."
fi

liste_fichiers=$(ls)
for fichier in $liste_fichiers
do
     echo "Fichier: $fichier"
done
```

Boucle `for` sur `$liste_fichiers`

Plan

Le système de fichiers

Les commandes importantes

Les scripts

Les références

Références

https://www.commentcamarche.net/contents/1143-unix-les-fichiers
https://community.jaguar-network.com
https://www.cs.upc.edu/~padro/Unixforpoets.pdf
https://doc.ubuntu-fr.org
https://www.funix.org/fr/unix/grep-find.htm
http://www-igm.univ-mlv.fr/~borie/cours/unix/cours1.pdf
http://lanterne-rouge.over-blog.org/
https://linux-actif.fr/uniq
http://www.linux-france.org/article/memo/node9.html
ftp://ftp.linux-france.org/pub/article/debutant/ini-unix.pdf
http://www.linux-france.org/article/sys/fichiers/fichiers-2.html
http://www.macformath.net/mpu/mosx/101/intro02.html
https://technique.arscenic.org/
https://www.tuteurs.ens.fr/unix/archives.html
https://www.tuteurs.ens.fr/unix/shell/entreesortie.html
http://tvaira.free.fr/esimed/unix/UNIX%20-%20TP.pdf
https://www.univ-orleans.fr/lifo/membres/Yannick.Parmentier/linux/support1.pdf
https://www.zebulon.fr/dossiers/tutoriaux/83-tutoriel-virtualbox-machine-virtuelle.html

“*Victoriae mundis et mundis lacrima*. Bon, ça ne veut absolument rien dire, mais je trouve que c’est assez dans le ton.”

- Le roi Loth

Contact :

pierre-antoine.jean