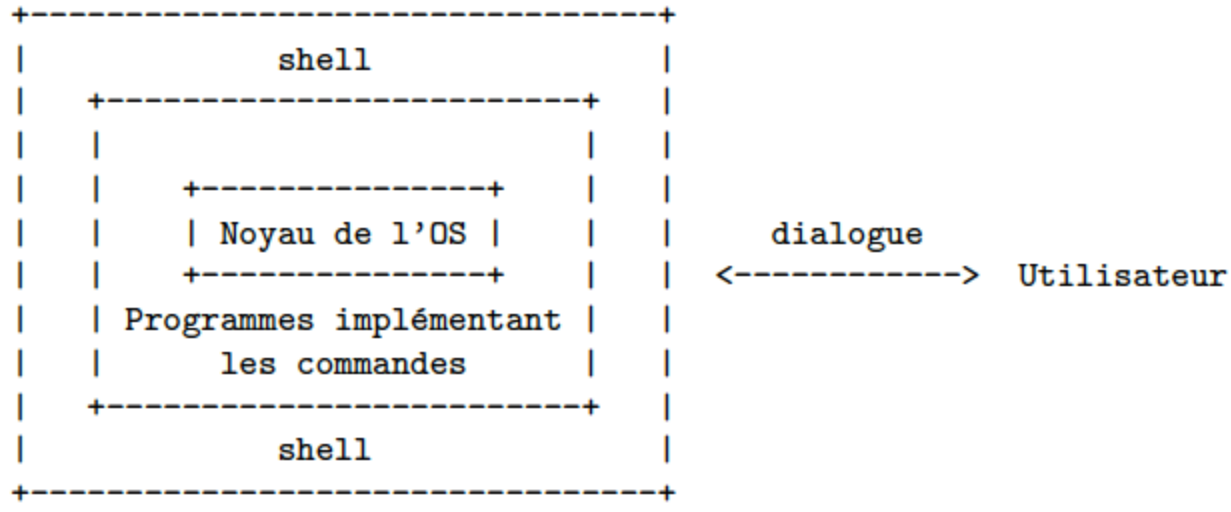
Shell

Qu'est ce qu'un shell ?

- Interface textuelle jouant le rôle d'interface entre l'utilisateur et la machine
- Etymologie :



- Bash, mais aussi plein d'autres (sh, csh, zsh, etc.)

##Au programme :

- Rappels sur l'arborescence et la désignation des fichiers
- Rappels sur les droits d'accès, les commandes courantes et quelques astuces
- Redirections des E/S
- grep et sed (et expressions régulières)
- Programmation Shell

Arborescence et fichiers

- Sous UNIX, une arborescence unique (cf. arbre)
- On exécute depuis le shell des commandes à un endroit spécifique de l'arborescence afin d'intéragir avec le système.
- Syntaxe des commandes : < commande > < arguments >. Les arguments sont séparés par des espaces, et les options avec -.

Commandes courantes :

- ls
- pwd
- cd
- cp
- mv
- mkdir
- rm

- Chaque fichier et répertoire : taille, date de creation, nom du propriétaire, droits d’accès
- Accessibles avec la commande ls -l.

```
-rw-r--r--   1 kevin  staff      11132 16 jan 10:13 bfs.py
```

- Droits d'accès : opérations permises par les utilisateurs en fonction de leur catégorie.
- 3 opérations : r, w, x
- 3 catégories : u, g, o (propriétaire, groupe, autres)
- Pour modifier ces droits : commande chmod

Exemples :

- chmod g+w fichier
- chmod o+rwx fichier

Question : Que signifient les droits d'écriture/lecture/éxécution pour les répertoires ?

Redirection des E/S

- Redirection des sorties : > et >>
- Redirection des entrées : <
- Tube de commandes : |

Question : Quelle est l'effet de la commande emacs fichier.c > fichier2.c ?

grep

- But : rechercher une expression régulière dans chaque ligne d'un ou plusieurs fichiers passés en paramètres (ou dans chaque ligne de l'entrée standard), et afficher ces lignes sur la sortie standard.

Options :

- -v : Lignes ne contenant pas d'occurrences
- -l : affichier le nom du fichier plutôt que les lignes
- -c : n'afficher que le nombre de ligne où l'occurrence apparaît.

Expressions régulières

Permettent de décrire des motifs formés de caractères.

Les expressions régulières sont construites à pratir de briques :

- Les expressions régulières atomiques
- L'association d'expressions régulières atomiques permettent de construire des expressions régulières simples
- Qui, associées, permettent de construire les expressions régulières étendues

Some people, when confronted with a problem, think : "I know, I'll use regular expressions."
Now they have two problems.
– Jamie Zawinski

Expressions régulières atomiques (era)

Motif d'un seul caractère.

- *ch* : le caractère *ch*
- \sp : le caractère spécial sp
- . : caractère quelconque
- [gkl] : un des caractères g, k ou l
- [^gkl] : un caractère autre que g, k ou l
- [a-z] : tout caractère entre a et z (en minuscules !)
- [^a-z] : tout caractère non compris entre a et z

Expressions régulières simples (ers)

Obtenues par concaténation d'expressions régulières atomiques ou par l'une des combinaisons :

- era* : tout mot de 0 à N caractères appartenant à era
- era+ : tout mot de 1 à N caractères appartenant à era
- era? : tout mot de 0 à 1 caractère appartenant à era
- era{n} : tout mot de n caractères appartenant à era
- era{n1,n2} : tout mot de n1 à n2 caractères appartenant à era

Expressions régulières étendues (ere)

- ^ere : la chaîne ere doit être en début de ligne
- ere$: la chaîne ere doit être en fin de ligne
- ers1|ers2 : tout chaîne qui vérifie ers1 ou ers2
- (ers) : toute chaîne qui vérifie ers
- (ers)* : concaténation de 0 à N chaînes vérifiants ers
- (ers)+
- (ers)?
- (ers){n}

Expressions régulières

- ◦ : répétition du caractère précédent, de longueur éventuellement vide (0 ou plus)
- ◦ : répétition du caractère précédent, de longueur non vide (1 ou plusieurs fois)
- ^ : début d'une ligne
- $: fin d'une ligne
- \ : caractère d'échappement, protège le caractère suivant
- {n} : répétition de n fois le caractère précédent
- [...] : caractères entre crochets, par énumération ou par intervalle.

Expressions régulières

Exercice.

sed

- Editeur de flux (Stream EDitor)
- Agit comme un filtre en lisant l'entrée standard ligne par ligne, puis en appliquant des commandes d'édition sur chacune.
- Le fichier d'origine est inchangé
- Le résultat est envoyé sur la sortie standard

sed

Fonctionnement :

- Chaque ligne du fichier d'netrée est copiée dans un buffer
- Toutes les commandes de sed sont appliquées, dans l'ordre, à ce buffer
- Le buffer est envoyé à la sortie standard

Principes fondamentaux :

- Toutes les commandes d'édition d'un script sed sont appliquées dans l'ordre à toutes les lignes de données (sauf restriction par adressage spécifique)
- Le fichier d'origine est inchangé.

sed

Syntaxe : *[adresse[,adresse]][!] commande [arguments]*

- Adresse : numéro de ligne ou expression régulière (exemples : 6, ou "/[1][0-9]/"
- Aucune adresse n'est indiquée : commande appliquée à chaque ligne du fichier.
- Une adresse : commande appliquée à chaque ligne correspondant.
- Deux adresses : commande appliquée à toutes les lignes entres ces deux adresses.
- ! : applique la commande sur toutes les lignes non désignées par cette adresse.

Exemple : '10,$ s/chat/chien' -> substitution de chat par chien de la ligne 10 à la fin du fichier.

sed

De nombreuses commandes, mais deux détaillées ici : s et d.

s : commande de substitution :

Syntaxe : *[adressage] s/motif/chaine_de_remplacement/[indicateurs]*

Indicateurs :

- n (1 <= n <= 512) : seule la nième occurrence du motif dans la ligne est remplacée
- g : toutes les occurrences
- p : contenu du buffer de travail

L'option *-n* supprime la sortie standard.

sed

\x (1<= x <= 9) permet de mémoriser uen portion de concordance du motif. Cette portion doit être entourée par (et) dans le motif.

Exemple :

s/ \(.*\) : \(.*\)/ \2 : \1/

Transforme le fichier :

- lundi : monday
- mardi : tueday

En :

- monday : lundi
- tuesday : mardi

sed

d : commande de destruction : si une ligne concorde, la ligne entière est détruite.

sed

Syntaxe globale de sed :

sed [-n] [-e script] [-f fichier_sed] [liste_de_fichiers]

sed : exercices

Que font les commandes suivantes ?

- sed -e 's/toto/tutu/g' fichier.txt
- sed -e '/^#/d' fichier.txt
- sed -e '1,10 s/toto/titi/g' fichier.txt
- sed -e '1,10 s/toto/titi/' fichier.txt
- sed -e '1,10 s/toto/titi/2' fichier.txt

Exercices