L' interpréteur de commandes (shell)

- Démarre après la connexion (login)
- Interprète et exécute les commandes tapées au clavier et les scripts
- Les shells: *sh, tch, bash,* ou autre ...
- Choix du shell par défaut

Configuration du shell (bash)

- Scripts communs à tous les utilisateurs
  - */etc/profile*
  - */etc/bash.bashrc* *(ou etc/bashrc)*
- Scripts cachés dans le répertoire utilisateur
  - */home/utilisateur/.profile* *(.bash_profile ou .bash_login)*
  - */home/utilisateur/.bashrc*
  - */home/utilisateur/.bash_logout*

Variables bash

- Créer, afficher, supprimer une variable locale
  - A=toto;
  - echo $A
  - unset A
- Augmenter la portée d'une variable
  - export $A
- Afficher les variables d'environnement
  - env pour les afficher toutes
  - $NOM_VARIABLE
- ~ est un raccourci pour $HOME

Entrée Sortie standard

- &0 stdin (clavier par défaut)
- &1 stdout (écran par défaut)
- &2 stderr (sortie erreur)

Enchaînement des commandes

- & Exécution en arrière-plan
- ; Séparateur
- # Commentaire
- | Pipe
- > Redirection de la sortie standard
- >> Redirection de la sortie standard en ajout
- < Redirection de l'entrée standard
- && Et
- || Ou

Scripts BASH

- Fichier texte qui contient une suite de commandes shell
- Utilisation de variables et de structures de contrôle
- **#! /bin/bash** première ligne
- Ne pas oublier:
  - Activer le droit x pour le ou les utilisateurs
  - Lancer le script en indiquant soit :
    - le chemin complet à partie de la racine
    - Le chemin relatif **./nomDuScript**
    - modifier la variable **$PATH** pour y inclure le répertoire ou se situe le script

Visibilité des variables utilisateurs

- Visibilité des variables
  Exemple:

  A=test PAS d'espaces!!!

  echo $A pour afficher le contenu de A

  export $A permet de rendre la variable A visible par un autre script (autre BASH)

Passage de paramètres

- Variables associées à un script:
  $0 le nom du script
  $1 à **$9** les paramètres qui suivent le nom
  $# nombre de paramètres
  $* la liste de tous les paramètres
  $? valeur de retour (0 si pas d'erreur)
  $$ numéro du PID courant
- Syntaxe:
  - Script *param1 param2 param3 ....*
    $0 $1 $2 $3 .....
- Utilisation de **shift**

Les tests

- ATTENTION: Pas d'erreur : 0 équivaut à vrai !!!
- 2 Syntaxes : test expression ou [expression]

- Tester un fichier *[option fichier]*
  option : *-e* existe *-d* répertoire *-s* pas vide
  -x exécutable *-w* modifiable *-r* lisible
- Tester si une chaîne est vide ou pleine *-z* ou *-n*
- Tester si deux chaînes sont == ou !=
- Tester deux nombres *-eq -ne -lt -gt -le -ge*
- *[expression1] -a [expression2]* ET logique
- *[expression1] -o [expression2]* OU logique
- [!expression] NEGATION

Stuctures conditionnelles

- Si..... sinon

```
if   commande ou test ; then
        instructions
else
        instructions
fi
```

- Imbrication possibles avec : *elif*
- Choix multiples:

```
case  valeur in
        expression1)      instructions  ;;
        expression2)      instructions  ;;
        ...
esac
```

Structures itératives

- *for variable in liste do*
  commandes utilisant $variable en général
  done

- *while test do*
  commandes
  done

- *Until test do*
  commandes
  done

Fonctions

- *nom_fonction() {*
  commandes
  }

- Autre syntaxe:

  fonction nom_fonction {
  commandes
  }