

	<p align="center"><i>Programmation Système</i> <i>C-shell & Processus</i></p>	<p align="center">Année universitaire : 2025/2026</p>
---	---	--

Objectifs :
<ul style="list-style-type: none"> • Créer des alias, complétion et historique • Comprendre le fonctionnement des processus

Exercice 1: Processus

Un processus est un programme qui s'exécute. Un processus a un propriétaire (l'utilisateur qui l'a lancé) et est identifié par un numéro : son pid (process identity). On peut visualiser les processus en cours grâce à la commande ps.

- 1.1 Utiliser la commande man pour connaître les options possibles de la commande ps et lancer la commande qui vous permet de visualiser tous les processus en cours avec l'utilisateur propriétaire. Le shell offre la possibilité de lancer des processus en arrière-plan (background) : l'utilisateur n'a pas besoin d'attendre la fin d'un programme pour commencer le suivant. Pour cela, on utilise le caractère &.
- 1.2 Lancer un emacs en premier plan depuis votre terminal de commande. Après l'avoir interrompu, trouver le numéro de job associé et son pid puis le relancer. On peut interrompre un processus en cours grâce à la commande Ctrl-Z. La commande jobs indique les processus en arrière-plan du shell en cours. On peut ensuite relancer en arrière-plan un programme interrompu en utilisant la commande bg suivie éventuellement du numéro fourni par jobs (la commande fg le relance en premier plan)
- 1.3 Lisez la page de manuel de la commande kill ainsi que la rubrique consacrée à kill dans le manuel de kill
- 1.4 Terminer le processus exécutant emacs en utilisant la forme de la commande kill qui utilise le numéro de job (pas le pid)
- 1.5 Interprétez la différence de fonctionnement du shell sur ces 2 commandes :
sleep 5
sleep 5 &
- 1.6 Que devez-vous faire pour invoquer textuellement (sans la souris) une autre fenêtre terminal ?
- 1.7 Lancez sleep 30 & puis listez vos processus dans l'autre (ps).
- 1.8 Récupérez l'identifiant (PID) du processus sleep puis terminez ce processus

Exercice 2 : Historique

Le C-shell dispose d'un mécanisme d'historique qui permet de retrouver les dernières commandes tapées. On peut en fixer la taille en fixant la valeur de la variable `history`. Par exemple, `set history = 100` permet d'enregistrer les 100 dernières commandes tapées dans le shell. On peut utiliser les flèches pour se déplacer dans l'historique ou bien utiliser le (meta-)caractère `!`. La commande `history` fournit l'historique des commandes tapées. La commande `!n` relance la commande numéro *n* et `!chaine` la dernière commande qui commence par la chaîne de caractères fournie.

2.1 Visualisez toutes les commandes que vous avez tapées depuis le début de la session

2.2 Rappeler des commandes en utilisant les deux mécanismes expliqués ci-dessus

2.3 Fixez votre taille d'historique à 20 commandes

2.4 Retapez `history`. Que constatez-vous ?

Exercice 3 : Complétion et Alias

Le `csh` est capable de compléter un mot à partir des premiers caractères. Pour cela, il faut entrer le début du mot (par exemple, `ls pr` puis taper sur la touche `tab`). Le shell essaie alors de compléter `pr`. Si plusieurs mots différents peuvent compléter ce que vous avez entré, le shell n'affichera rien. Si vous tapez alors `Ctrl-D`, toutes les possibilités seront affichées. Vous pourrez alors entrer les caractères supplémentaires nécessaires qui permettront de choisir entre les diverses possibilités.

Le C-shell permet de créer des alias des commandes. Le but est soit de simplifier une commande ou de forcer une option de commande. Un alias est créé grâce à la commande `alias nom commande`. On le supprime avec `unalias nom`.

3.1 Utiliser la complétion pour afficher toutes les commandes du répertoire `/usr/sbin` qui commencent par `ch`.

3.2 Créer les alias suivants

- `m` pour `more`
- `del` pour `rm -i`
- `ll` pour `ls -al`
- `lr` pour une commande qui liste récursivement le contenu d'un répertoire

Exercice 4: Métacaractères

Les métacaractères sont un mécanisme très puissant des systèmes Unix. Les principaux métacaractères sont rappelés ici.

- `*` remplace zéro fois ou autant de fois que voulu n'importe quel caractère.
- `?` remplace exactement un caractère.
- `^` désigne un début de ligne.
- `$` désigne une fin de ligne.

- \ indique que caractère suivant doit être interprété normalement.
- & permet de lancer une commande en arrière-plan.
- ; permet de séparer des commandes écrites sur une seule ligne.
- [] permettent d'identifier une liste ou un intervalle de caractères. Par exemple ls [a-p]* liste tous les fichiers du répertoire courant qui commencent par une minuscule comprise entre 'a' et 'p'.

4.1 Écrire les commandes permettant de lister dans les répertoires de votre choix

- tous les sources de programme C++ (se terminant par .cc) ;
- les fichiers commençant par fic ;
- les programmes C++ commençant par tp ;
- les fichiers de la forme tpx.c ou x est un chiffre (de 0 à 9) ;
- les fichiers qui commencent par une majuscule comprise entre 'B' et 'I' ;
- les fichiers dont le nom comporte au moins un caractère compris entre 'p' et 'w' ;
- les fichiers dont les noms se terminent par une minuscule comprise entre 'l' et 'p' ;
- les fichiers dont le nom commence par une minuscule comprise entre 'b' et 'f' et ceux dont le nom commence par une minuscule comprise entre 'r' et 'u'.