

Mémo Linux

Table des matières

1	Notions de base	3
2	Navigation	3
2.1	cd	3
2.2	pwd	3
3	Fichiers et dossiers	3
3.1	ls	3
3.2	mv — Déplacer ou renommer	4
3.3	cp — Copier	4
3.4	rm — Supprimer	4
3.5	touch — Créer des fichiers vides	4
3.6	mkdir — Créer des dossiers	4
4	Lecture de fichiers et recherche	4
4.1	cat — Afficher le contenu	4
4.2	grep — Rechercher du texte	5
5	Redirection	5
6	Éditeur de texte	5
6.1	nano	5
6.2	Vim	5
6.2.1	Lancer Vim	5
6.2.2	Modes de Vim	5
6.2.3	Commandes :	6
7	Gestionnaire de fenêtres : i3	6
8	Git	6
8.1	git config — Configurer Git	6
8.2	git status — Vérifier l'état du dépôt	6
8.3	git add — Sélectionner les modifications	7
8.4	git commit — Créer un commit	7
8.5	git log — Historique	7
8.6	Branches	7
8.7	Remote — Dépôts distants	7
8.8	git tag — Marquer une version	7
8.9	Explication des tags <code>archi</code> et <code>submit</code>	7
8.10	Bonnes pratiques	8

9	Cloner les TP depuis la Forge	8
9.1	Accéder au tp	8
10	Ajouter une clé SSH	9
10.1	Génération de clé SSH selon votre système	9
10.1.1	Linux	9
10.1.2	Mac	9
10.1.3	Windows	9
10.2	Ajouter la clé à la Forge	9
10.3	Si vous avez déjà une clé SSH	9

Introduction

Ce mémo contient les **commandes Linux les plus utiles** pour l'année et plus encore. *N'hésitez pas à demander si vous voulez que je rajoute une commande.*

1 Notions de base

- **Racine (/)** : le dossier le plus haut de l'arborescence.
- **Home (/home)** : contient tous les dossiers personnels des utilisateurs.
- **Dossier personnel (~)** : votre espace à vous.
- **AFS** : votre répertoire de travail personnel (dans votre home).
- **cmd** : l'invite de commande que vous ouvrez avec **mod+Enter**.
- **Moulinette** : programme qui note les TP en pourcentage sur 100.

2 Navigation

2.1 cd

```
cd ..          # remonter d'un dossier
cd ../../..    # remonter de plusieurs niveaux
cd algo        # entrer dans le dossier algo
cd algo/ocaml/TP # aller directement dans TP
```

Autocomplétion : `cd an<TAB>` complète automatiquement (anglais) `cd a<TAB><TAB>` affiche toutes les possibilités (algo/ anglais/)

2.2 pwd

```
pwd # affiche le chemin complet du dossier courant
```

3 Fichiers et dossiers

3.1 ls

```
ls          # liste les fichiers du dossier courant
ls -l       # liste détaillée
ls -a       # inclut les fichiers cachés
```

3.2 mv — Déplacer ou renommer

```
mv [origine] [destination] # déplacer ou renommer
mv fichier nouveau         # renommer
mv anglais/* algo          # déplacer tout
mv A* anglais              # déplacer tous les fichiers
    commençant par A
```

3.3 cp — Copier

```
cp [source] [destination] # copier un fichier
cp -r dossier/ copie_dossier/ # copier un dossier
```

3.4 rm — Supprimer

```
rm fichier.txt           # supprimer un fichier
rm -rf dossier/          # supprimer un dossier et son contenu
rm fichier1 fichier2     # supprimer plusieurs fichiers
rm -rf a*                # supprimer tous les fichiers/dossiers
    commençant par "a"
```

Attention : ne jamais utiliser `rm -rf` sur `/`, votre home ou AFS!

3.5 touch — Créer des fichiers vides

```
touch test.txt
touch f1.txt f2.cs f3
```

3.6 mkdir — Créer des dossiers

```
mkdir algo/perl
mkdir dossier1 dossier2 dossier3
```

4 Lecture de fichiers et recherche

4.1 cat — Afficher le contenu

```
cat fichier.txt          # affiche le contenu du fichier
```

4.2 grep — Rechercher du texte

```
grep "mot" fichier.txt      # recherche "mot" dans le fichier
grep -r "mot" dossier/      # recherche récursive dans un
                             dossier
```

5 Redirection

```
commande > fichier          # copie la sortie dans fichier ( écrase
                             le contenu)
commande >> fichier         # ajoute la sortie à la fin du fichier
```

6 Éditeur de texte

6.1 nano

```
nano fichier.py
```

Commandes :

- Ctrl+S : enregistrer
- Ctrl+X : quitter
- Ctrl+Shift+C : copier
- Ctrl+Shift+V : coller

6.2 Vim

6.2.1 Lancer Vim

```
vim fichier.txt
```

6.2.2 Modes de Vim

- **Mode normal** : par défaut, permet de naviguer et lancer des commandes.
- **Mode insertion** : permet d'écrire du texte. (touche i)
- **Mode commande** : permet d'exécuter des ordres (touche : depuis le mode normal).

6.2.3 Commandes :

```
i      # entrer en mode insertion
ESC    # revenir en mode normal
:w     # enregistrer
:q     # quitter
:wq    # enregistrer et quitter
:q!    # quitter sans enregistrer
dd     # supprimer une ligne
yy     # copier une ligne
p      # coller
```

7 Gestionnaire de fenêtres : i3

Les salles machines utilisent **i3**, un gestionnaire de fenêtres (vous pouvez aussi l'installer chez vous).

Raccourcis principaux (touche Mod = Windows ou Alt)

- Mod+Enter : ouvrir un terminal.
- Mod+d : ouvrir le menu des applications.
- Mod+Shift+q : fermer une fenêtre.
- Mod++/+/-/↑/↓ : se déplacer entre les fenêtres.
- Mod+Shift++/+/-/↑/↓ : déplacer une fenêtre.
- Mod+1..9 : changer d'espace de travail.
- Mod+Shift+1..9 : envoyer la fenêtre courante vers un autre espace.

Attention : toujours **i3-lock** votre session quand vous n'êtes pas devant même si vous êtes dans la salle !

8 Git

8.1 git config — Configurer Git

```
git config --global user.name "Ton Nom"
git config --global user.email "ton@mail.com"
```

8.2 git status — Vérifier l'état du dépôt

```
git status    # montre les fichiers modifiés, ajoutés ou suivis
```

8.3 git add — Sélectionner les modifications

```
git add fichier.txt      # ajoute un fichier
git add .                # ajoute tous les fichiers modifiés
```

8.4 git commit — Créer un commit

```
git commit -m "message clair"
```

8.5 git log — Historique

```
git log                  # affiche l'historique complet
git log --oneline --graph # affichage condensé et graphique
```

8.6 Branches

```
git branch              # liste les branches
git checkout -b dev     # créer et aller sur une nouvelle branche
git switch dev          # basculer sur la branche dev
git merge dev           # fusionner dev dans la branche courante
```

8.7 Remote — Dépôts distants

```
git pull origin main    # récupérer les modifs
git push origin main     # envoyer les modifs
```

8.8 git tag — Marquer une version

```
git tag -ma <TAG>       #Le tag doit etre soit archi- soit submit-
git push --follow-tags   # envoyer le tag
```

Attention : Ne jamais submit avant un archi, vous prenez le risque d'avoir un 0%

8.9 Explication des tags archi et submit

- **Tag archi** : Permet de vérifier que votre code compile, qu'il n'utilise pas de fonctions interdites et que l'arborescence de votre dépôt est correcte (pas de fichiers en trop ou manquants).
- **Tag submit** : Déclenche la notation par la moulinette. Attention : si une fonction interdite est détectée, le code ne sera pas corrigé et la note sera de **0%**.

8.10 Bonnes pratiques

- Toujours faire un `git status` avant de commit ou push.
- Messages de commit clairs et courts.
- Toujours faire un tag archi avant un submit
- Ne jamais commit de fichiers inutiles (*.tmp, *.o, etc.) → utiliser `.gitignore` (Il est donné dans chaque TP).

9 Cloner les TP depuis la Forge

9.1 Accéder au tp

1. Rendez-vous sur la Forge : <https://cri.epita.fr/>
2. Cliquez sur **Intranet Forge**.
3. Allez dans **Cycle préparatoire - Computer Science**.
4. Cliquez sur le premier rectangle.
5. Cliquez ensuite sur **PROG** (grand rectangle).
6. Copiez le lien `git remote`, par exemple :

```
prenom.nom@git.forge.epita.fr:p/epita-prepa-computer-science/  
prog-101-p-01-2030/epita-prepa-computer-science-prog-101-p  
-01-2030-prenom.nom.git
```

7. Allez dans votre dossier afs et tapez :

```
git clone <CTRL+SHIFT+V pour coller le lien>
```

8. Entrez dans le dossier cloné :

```
cd epita-prepa-computer-science-prog-101-p-01-2030-prenom.nom
```

9. Le sujet se trouve dans la section **Subject** sur la Forge.

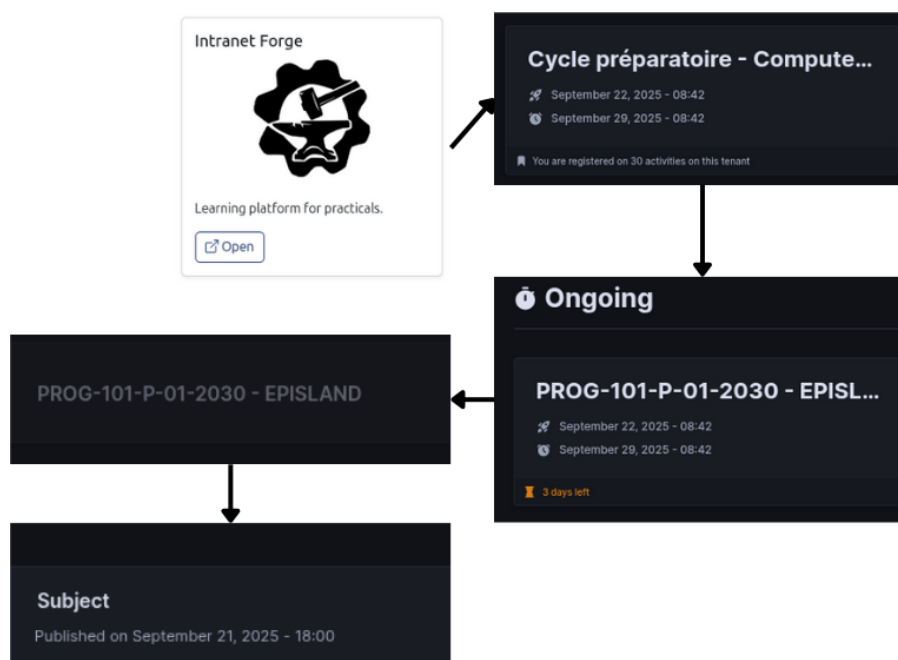


FIGURE 1 – Chemin pour un TP

10 Ajouter une clé SSH

10.1 Génération de clé SSH selon votre système

10.1.1 Linux

Linux (Debian, Ubuntu, Arch, Fedora, ...)

1. Ouvrez un terminal.
2. Tapez la commande : `ssh-keygen -a 100 -t ed25519`
3. La clé sera générée dans `~/.ssh/id_ed25519` (privée) et `~/.ssh/id_ed25519.pub` (publique).

10.1.2 Mac

macOS

1. Ouvrez le terminal (Applications → Utilitaires → Terminal).
2. Utilisez la même commande que sous Linux : `ssh-keygen -a 100 -t ed25519`
3. Les fichiers seront créés dans `~/.ssh/`.

10.1.3 Windows

Windows (PowerShell)

1. Ouvrez PowerShell.
2. Tapez la commande : `ssh-keygen -t ed25519 -a 100`
3. Les fichiers seront créés dans `C:\Users\VotreNom\.ssh`.

10.2 Ajouter la clé à la Forge

1. Ouvrez le fichier `id_ed25519.pub` avec un éditeur de texte (ou utilisez `cat` sous Linux/macOS).
2. Copiez son contenu.
3. Rendez-vous sur la Forge : <https://cri.epita.fr>
4. Cliquez sur votre profil (en haut à droite).
5. Allez dans la section **SSH KEYS**.
6. Collez la clé copiée dans le champ prévu.
7. Donnez un nom explicite (par ex. `laptop-linux`, `pc-windows`, `macbook`) pour savoir à quelle machine correspond cette clé.

10.3 Si vous avez déjà une clé SSH

1. Ouvrez le dossier `.ssh` :
 - Linux/macOS : `~/.ssh`
 - Windows : `C:\Users\VotreNom\.ssh`
2. Copiez le contenu du fichier se terminant par `.pub`.
3. Ajoutez-la à la Forge de la même manière que ci-dessus.