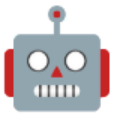

Cours 4 - 11 Octobre

Date de création	@10 octobre 2023 21:06
Cours	Programmation et Projet Encadré

Correction

Exercice 1

```
micro journal.md
git diff #voir les modifications
git add journal.md
git commit -m "I AM ERROR"
git log #pour récupérer le SHA du commit à supprimer
git revert <commit> #pour retirer ce commit #revert à utiliser si push
git reset HEAD~1 #si pas push
git pull
```

Exercice 2

```
micro oups.md
git add oups.md
git commit -m "maj journal"
micro oups.md
git add oups.md
git commit -m "maj2 journal" #faire plusieurs "micro, add, commit"
git tag gitmoreinit -a -m "Mon premier tag"
git reset gitmoreinit
#on copie-colle le contenu de oups.md dans journal.md
git rm oups.md
git commit -m "suppression oups.md"
git push
```

git reset → on reviens dans le temps

git revert → prend un commit et on le supprime

Cours

PENSER A LANCER BASH SUR MAC

> 00-Shell.pdf

^+D → permet de dire qu'on a finit de taper quelque chose

< : remplace le clavier par le contenu d'un fichier

> : écrit stdout dans un fichier

2> : écrit stderr dans un fichier
>& : écrit stdout et stderr dans un fichier

En écriture, si on double le chevron (>>, >>&, 2>>), on écrit en ajoutant la sortie à la fin d'un fichier.
ATTENTION: les chevrons simples (>, >&, 2>) écrasent le fichier si il existe déjà.

wc < fic.txt → prendre comme entrée fic.txt, va compter le nombre de caractère dans fichier.txt

> wc < fic.txt → il y a une redirection, marque la fin de la commande
>
> wc fic.txt → on donne fic.txt comme argument

wc fic.txt > output.txt → met ce qu'on obtient dans un fichier de sortie output.txt

`cat *.txt | grep université | wc > output.txt` va lire tous les fichiers .txt, on va ensuite utiliser grep pour retourner toutes les lignes de texte où se trouve "université" et ensuite on va compter les lignes, mots et caractères et le résultat va être envoyer dans un fichier de sortie output.txt

grep : recherche de motifs dans l'entrée (ou dans des fichiers)
sort : trier des lignes

uniq : supprimer les lignes qui se répètent
echo : affiche un texte (pour formater vos résultats)

• Compter le nombre d'annotations (nombre de lignes) par année (2016, 2017 et 2018)

```
cat ./2016/*/*.ann | wc -l
```

à faire pour 2017 et 2018

• limiter ce comptage aux lieux (Location)

```
cat ./2016/*/*.ann | grep Location | wc -l
```

• sauvegarder ces résultats dans un (seul) fichier

```
echo "Locations en 2016" > reponse.txt
cat ./2016/*/*.ann | grep Location | wc -l >> reponse.txt
echo "Locations en 2017" >> reponse.txt
cat ./2017/*/*.ann | grep Location | wc -l >> reponse.txt
echo "Locations en 2018" >> reponse.txt
cat ./2018/*/*.ann | grep Location | wc -l >> reponse.txt
```

• établir le classement des lieux les plus cités.

```
cat ./2016/*/*.ann | grep Location | cut -f 3 | head
```

le -f : pour dire qu'on sélectionne la colonne

Le 3 désigne la troisième colonne

head : affiche les 10 premières lignes seulement

si on veux les 15 premières lignes on ajoute après `head -n 15`

```
cat ./2016/*/*.ann | grep Location | cut -f 3 | head | sort | uniq -c
```

sort : trier dans l'ordre alphabétique

uniq : supprime les doublons

-c : permet de compter

```
cat ./2016/*/*.ann | grep Location | cut -f 3 | sort | uniq -c | sort -nr | head -n 20
```

sort -nr : permet de trier par ordre décroissant en fonction du nombre (-n permet de trier numériquement ; r : ordre décroissant)

• trouver les annotations les plus fréquentes pour votre mois de naissance, toutes années confondues.

```
cat ./*/02/*.ann | grep Location | cut -f 3 | sort | uniq -c | sort -nr | head -n 20
```

écrire un script bash:

```
vim test.sh #on écrire le script
chmod +x test.sh #rend le script executable
./test.sh #execute le script
```

Quand on affecte quelque chose à une variable on ne met pas de $, mais quand on veux faire référence à la variable on ajoute $ devant le nom de la variable.

```
NB=$(cat *ann | wc -l)
echo "nombre d'annotations : $NB "
```