TP3 : Gérer les commits avec Git


```
Ce TP est une suite du TP1 et TP2. Pour le réaliser, vous avez le choix entre le
faire en ligne de commande ou à l'aide
d'une IDE (VisualStudioCode ou une suite JetBrains).

L'objectif de ce TP est de vous familiariser à la manipulation d'historique des
commits (renommage, suppression de fichiers,
rebases interactifs), gérer des erreurs dans des branches et nettoyer un dépôt de
manière professionnelle.

⚠ N'oubliez pas de joindre des captures d'écrans et d'expliquer ce que vous
voyez à l'écran.
```

- **Question 1.0 :** Dans le premier TP, vous avez récupéré une branche sur Gitea. Ce qu'on ne vous a pas dit, c'est qu'il y a deux commits qui **ne mentionnent pas l'existence d'un fichier**. Vous allez devoir **trouver** et **revenir** sur ce commit

Avec la commande suivante, on devrait voir les commits détaillés avec les ajouts/suppressions de lignes qu'il y a eu sur un fichier.

```
git log --oneline --stat
```

Ensuite, grâce à la commande précédente on peut se placer sur le commit où le fichier a été créé.

```
git checkout <commit-id>
```

- **Question 1.1 :** Maintenant que vous avez récupéré le fichier, **utilisez** le fichier en question et **adaptez-le avec votre nom de famille** afin d'obtenir un résultat. N'hésitez pas à spécifier ce que vous avez mis comme input et ce que vous avez reçu comme output.

Dans le fichier temp.py, il faudra que l'élève l'adapte avec son nom de famille :

```
def chiffrement(nomdefamille):
    """
    :param nomdefamille: Votre nom de famille tout en MAJUSCULE
    :return: un entier représentant la somme des codes ASCII de chaque lettre du
nom de famille
    """
    return sum(ord(char) for char in nomdefamille)
```

```
if __name__ == "__main__":
    resultat = chiffrement("TEST")
    print(resultat)
```

⚠ À partir de la question 1.2, le support de cours ne couvre plus le TP3. Ce sera à vous de faire les recherches là-dessus et d'expliquer ce que vous avez compris.

- **Question 1.2 :** Pour corriger cette erreur, vous allez maintenant **renommer** les commits qui ne mentionnaient pas le nom du fichier que vous avez trouvé. Puis poussez la branche en utilisant `git push --force` afin d'écraser la précédente qui avait des commits incorrectement nommés.

```
git rebase -i HEAD~N //Remplacer N par le nombre de commit à modifier
git push --force
```

- **Question 2 :** Créez une nouvelle branche avec un nom judicieux de votre choix, basculez sur cette branche puis créez un fichier nommé **gestion_nom.py**

```
git branch <nom-de-la-branche>
git checkout <nom-de-la-branche>
```

Création d'un fichier gestion_nom.py

- **Question 3.0 :** Dans ce fichier, créez une fonction qui ajoute un nom à une liste de nom. Versionnez vos changements. L'élève fera le code, le code n'a pas forcément besoin d'être correcte.

```
git add gestion_nom.py
git commit -m "Ajout de la fonction ..."
```

- **Question 3.1 :** Ajoutez une nouvelle fonction qui vérifie si le nom est déja présent dans la liste si c'est le cas, il n'est pas ajouté. Versionnez vos changements. L'élève fera le code, le code n'a pas forcément besoin d'être correcte.

```
git add gestion_nom.py
git commit -m "Ajout de la fonction ..."
```

- **Question 3.2 :** Ajoutez une nouvelle fonction qui permet de supprimer un nom de la liste. Mettez en ligne cette version
- L'élève fera le code, le code n'a pas forcément besoin d'être correcte.

```
git add gestion_nom.py
git commit -m "Ajout de la fonction ..."
```

- **Question 3.3 :** Ajoutez une nouvelle fonction qui permet de trier par ordre Alphabétique les noms de la liste. Mettez en ligne cette version L'élève fera le code, le code n'a pas forcément besoin d'être correcte.

```
git add gestion_nom.py
git commit -m "Ajout de la fonction ..."
```

- **Question 4 :** Supprimez un par un les 4 commits précédant créés dans la branche, puis supprimez la branche elle-même. Attention, vous devez documenter et illustrer vos résultats à chaque étape, il ne sert à rien de supprimer directement la branche.

```
git rebase -i HEAD~4
```

À la place des "pick" remplacer par "drop" tel que :

```
drop 1234567 Commit 1
drop 2345678 Commit 2
drop 3456789 Commit 3
drop 4567890 Commit 4
```

Pour supprimer la branche

```
git branch -d <nom-de-la-branche>
```

Travail facultatif : Afin de vous familiariser davantage avec l'outil graphique Git intégré dans JetBrains, vous pouvez refaire les TPs, mais cette fois en utilisant l'interface graphique de JetBrains et en joignant des captures d'écrans des actions réalisées.