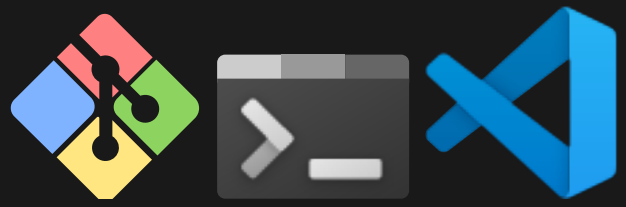
Debrief séance 1

UE12 P24 - Python

Configuration

1. Où est Python ?
2. Où écris-je mon code ?
3. Quel est mon dossier de travail ?
4. Mon IDE est-il bien configuré ?

Outils



Raccourcis utiles

- Tous les terminaux :
  - ↑ : commande précédente
  - ↓ : commande suivante
- Windows Terminal :
  - Alt + Maj + D : scinde le terminal en 2
  - Ctrl + Maj + W : ferme le terminal actuel
- VS Code :
  - Ctrl + K, Ctrl + C : commente la sélection
  - Ctrl + K, Ctrl + U : décommente la sélection

Vocabulaire

- Indentation : espaces avant le début d'une ligne
  - Python s'en sert pour délimiter les blocs de code (fonctions, boucles, conditions)
  - Ça aide la lisibilité

```
def user_list_
    print()
    print('Use
    print('---
    print()
    for index,
        print(
    print()
    print('n.
```

- Refactoring ("réusinage" pour les Québecquois) : réorganisation du code pour améliorer lisibilité et maintenabilité

Un peu de Python

- Les f-string, une façon lisible d'afficher des variables dans une string :

```
1 user = {'id': 41, 'name': "Alice"}
2 print(f'Hello {user['name']}! Your id is {user['id']}.')
```

```
Hello Alice! Your id is 41.
```

- La bibliothèque standard : si vous cherchez un algorithme, il est sûrement déjà codé quelque part

```
1 new_channel_users_string = 'Alice, Bob, Charlie'
2 new_channel_users_string.split(',')
```

```
['Alice', ' Bob', ' Charlie']
```

```
1 [user.strip() for user in new_channel_users_string.split(',')]
```

```
['Alice', 'Bob', 'Charlie']
```

Flot de contrôle

Python lit les lignes de votre programme les unes après les autres :

```
1 display_main_menu()
2 choice = input('Continue ? (Yes/No): ')
3 print(choice)
```

Si on veut juste recommencer la première action, on peut utiliser une boucle :

```
1 choice = ''
2 while choice != 'No':
3     display_main_menu()
4     choice = input('Continue ? (Yes/No): ')
```

Flot de contrôle - Fonctions

Si on veut plus de flexibilité, on fait des fonctions :

```
1 def display_main_menu():
2     choice = input('Continue ? (Yes/No): ')
3     display_user_menu()
4
5 def display_user_menu():
6     choice = input('Continue ? (Yes/No): ')
7     display_main_menu()
8
9 display_main_menu()
```

Façon de travailler - Découpage

Si un problème est compliqué, on le découpe en sous-tâches plus simples :

```
1 user_list.append({'id': new_id, 'name': username})
```

devient

```
1 new_user = {'id': new_id, 'name': username}
2 user_list.append(new_user)
```

Façon de travailler - Refactoring

Une fois le problème résolu, on refactore :

```
1 print(users[0])
2 print(users[1])
```

devient

```
1 for user in users:
2     print(user)
```

> **Tip**
>
> Ne cherchez pas à optimiser prématurément

Git - Une habitude à prendre

L’essentiel : utiliser Git dès que possible.

- Création d’un dossier dédié au projet
- `git init`

Puis dès qu’une nouvelle fonctionnalité est développée :

- `git add`
- `git commit`