



ENSGTI
ÉCOLE D'INGÉNIEURS



Programmation Orientée Objet (Python)

TP1 : Bases du langage et environnements virtuels

[A.Martin](#)

2026-2027



Visual Studio Code



Objectif

Revoir ou découvrir les bases du langage Python et apprendre à utiliser les environnements virtuels.

Durée estimée : 4 heures

1. Introduction

L'objectif de ce premier TP est double. Tout d'abord, vous découvrirez ou réviserez les bases du langage Python : variables, types, structures de contrôle, structures de données, fonctions, fichiers, exceptions. En effet, même si vous avez déjà développé en Pascal ou dans d'autres langages, Python présente des spécificités qu'il est indispensable de maîtriser. Ensuite, vous vous initiez à la notion d'environnement virtuel. À dire vrai, chaque projet Python devrait être isolé dans un environnement spécifique afin d'éviter tout conflit avec les autres projets.

2. Exercices

Exercice n° 1 : Terminal et programme

1. Ouvrez un terminal.
2. Créez le dossier « tp_python » dans lequel vous vous déplacerez (commandes `mkdir` et `cd`).
3. Récupérez le fichier ressources « tp1_basics.zip » sur Moodle, placez-le dans le dossier « tp_python » et désarchivez-le avec la commande suivante :

```
unzip tp1_basics.zip
```

4. Déplacez-vous dans le dossier « tp1_basics » qui devrait avoir l'arborescence suivante :

```
tp1_basics/  
|-- data/  
|   |-- capteur_temp.txt  
|   |-- capteur_press.txt  
|   |-- capteur_hum.py  
|-- src/  
|   |-- bonjour.py  
|-- README.md  
|-- requirements.txt
```

5. Visualisez le contenu du programme « bonjour.py » à l'aide de la commande :

```
more src/bonjour.py
```

6. Lancez le programme à l'aide de la commande suivante :

```
python src/bonjour.py
```

Exercice n° 2 : Premier environnement virtuel

1. Dans le dossier « tp1_basics », créez l'environnement virtuel « env_tp1 » puis constatez la nouvelle arborescence :

```
tp1_basics/  
|-- data/  
|   |-- capteur_temp.txt  
|   |-- capteur_press.txt  
|   |-- capteur_hum.py  
|-- env_tp1/  
|   |-- ...  
|-- src/  
|   |-- bonjour.py
```

```
|-- README.md  
|-- requirements.txt
```

2. Activez l'environnement virtuel et affichez la version de Python utilisée.

3. Installez les bibliothèques depuis le fichier « requirements.txt » :

```
pip install -r requirements.txt
```

4. Installez la bibliothèque « colorama » qui permet des affichages colorés sur le terminal, et qui n'était pas présente dans le fichier « requirements.txt » :

```
pip install colorama
```

5. Vérifiez son installation :

```
pip list
```

6. Désactivez l'environnement virtuel.

Exercice n° 3 : Programme utilisant la bibliothèque

1. Lancez l'éditeur Visual Studio Code (commande code) et depuis l'interface choisissez d'ouvrir le dossier « tp1_ basics ». Dans la partie gauche, vous devriez voir les sous-dossiers « data », « env_tp1 », et « src ». Créez un nouveau fichier Python « bonjour_couleur.py » dans le sous-dossier « src ».

2. Dans ce fichier, écrivez le code suivant :

```
from colorama import Fore, Style  
print(Fore.CYAN + "Bonjour l'ENSGTI!" + Style.RESET_ALL)
```

3. Visual Studio Code dispose d'un terminal intégré. Lorsque vous cliquez sur le bouton run il active automatiquement l'environnement virtuel. Testez l'affichage couleur.

4. Vérifiez qu'il fonctionne uniquement si l'environnement virtuel est activé. Pour faire ceci, vous pouvez désactiver l'environnement virtuel puis sélectionner l'interpréteur Python 3 en bas à droite de la fenêtre de VSCode.

Exercice n° 4 : Organisation de projet

1. Vérifiez que votre dossier de travail respecte cette fois-ci l'arborescence suivante :

```
tp1_basics/  
|-- data/  
|   |-- capteur_temp.txt  
|   |-- capteur_press.txt  
|   |-- capteur_hum.txt  
|-- env_tp1/  
|   |-- ...  
|-- src/  
|   |-- bonjour.py  
|   |-- bonjour_couleur.py  
|-- README.md  
|-- requirements.txt
```

2. Vérifiez que votre environnement virtuel est activé et exportez à nouveau les bibliothèques utilisées dans l'environnement virtuel :

```
pip freeze > requirements.txt
```

3. Supprimez l'environnement à l'aide de la commande :

```
rm -r -Force env_tp1
```

Et vérifiez que le dossier « env_tp1 » a bien disparu.

4. Recréez alors l'environnement avec les commandes :

```
python -m venv env_tp1
env_tp1\Scripts\activate
pip install -r requirements.txt
```

5. Vérifiez que les programmes fonctionnent toujours.

Exercice n° 5 : Manipulation des structures de données avancées

1. Listes

Créez le programme « liste.py » dans le sous-dossier « src ». Dans celui-ci, vous allez :

1. Créer une liste nommée mes_temperatures contenant les valeurs suivantes : 23.5, 24.0, 22.8, 23.1.
2. Ajouter la valeur 24.3 à la fin de la liste.
3. Afficher la température maximale, minimale et la moyenne. Testez le programme.

2. Tuples

Créez le programme « tuple.py » dans le sous-dossier « src ». Dans celui-ci, vous allez :

1. Créer un tuple nommé coordonnees_capteur contenant les valeurs (45.76, 4.84) représentant la latitude et la longitude.
2. Afficher ces coordonnées en les formatant dans une chaîne de caractères. Testez le programme.

3. Dictionnaires

Créez le programme « dictionnaire.py » dans le sous-dossier « src ». Dans celui-ci, vous allez :

1. Créer un dictionnaire nommé capteur avec les clés et valeurs suivantes :
 - 'id' : 'C001'
 - 'type' : 'Température'
 - 'valeur' : 23.5
2. Mettre à jour la valeur du capteur à 24.0.
3. Ajouter une nouvelle clé 'unité' avec la valeur '°C'.
4. Afficher les informations du capteur sous forme de phrase. Testez le programme.

4. Ensembles

Créez le programme « ensemble.py » dans le sous-dossier « src ». Dans celui-ci, vous allez :

1. Créer un ensemble nommé materiaux_detectes contenant les éléments 'Uranium', 'Plutonium', 'Thorium', 'Uranium'.
2. Afficher l'ensemble et noter le comportement concernant les doublons.
3. Ajouter le 'Césium' à l'ensemble. Testez le programme.

Exercice n° 6 : Manipulation de fichiers et gestion des exceptions

1. Un fichier nommé « capteur_temp.txt » dans le dossier « data » contient des mesures de température (une valeur par ligne). Écrivez un programme qui ouvre ce fichier et lit toutes les valeurs, et qui calcule :
 - La température moyenne.
 - La température maximale et minimale.
2. Gestion des exceptions
 - Gérez l'exception si le fichier n'existe pas en affichant un message d'erreur clair.
 - Gérez les exceptions liées à la conversion de chaînes en nombres (par exemple, si une ligne du fichier n'est pas une valeur numérique).
3. Affichez les résultats avec deux chiffres après la virgule.

Exercice n° 7 : Analyse des données de capteurs multiples

Dans le dossier « data », on dispose de trois fichiers contenant les données de différents capteurs : « capteur_temp.txt », « capteur_press.txt » et « capteur_hum.txt ».

Chaque fichier contient des mesures prises au même intervalle de temps.

1. Écrivez un programme qui :

- Lit les données des trois fichiers,
- Crée un dictionnaire où chaque clé est un horodatage et la valeur est un autre dictionnaire contenant les mesures des trois capteurs,
- Affiche les mesures combinées pour chaque horodatage.

2. Utilisez le numéro de ligne comme horodatage.

3. Si les fichiers n'ont pas le même nombre de lignes, prenez le minimum pour éviter les erreurs d'index.

4. Gérez les exceptions liées à la conversion de valeurs en float.

Résumé

Dans ce TP, vous avez découvert ou revu les concepts fondamentaux du langage Python ainsi que les structures de données avancées. Ces bases vous permettront d'aborder pleinement les séances suivantes qui introduiront des concepts plus avancés en programmation. D'autre part, vous avez mis en œuvre un environnement virtuel Python dans un cadre donné, ici une séance de TP. Ces environnements virtuels vous permettront d'isoler vos projets par la suite et ainsi d'éviter tout conflit entre eux.

Pour aller plus loin

- Documentation : Consultez la [documentation officielle de Python](#) pour approfondir chaque sujet.
- Tutoriels : Des plateformes comme [Un zeste de Python](#) et [Real Python](#) offrent des tutoriels détaillés sur divers sujets Python.