

À faire individuellement ou par petits groupes de deux ou trois.

## Exercice 1. Remarques préliminaires et configuration initiale

Ici, vous allez configurer votre machine de travail pour faire les séries d'exercices du semestre.

*Si vous utilisez l'infrastructure des machines virtuelles de l'EPFL:*

### Connexion

Connectez-vous sur l'infrastructure des postes de travail virtuels:

- en salle BC07-08: identifiez-vous directement sur les machines avec votre identifiant GASPARG;
- en salles CM1103 et CM1112: ouvrez une session sous Windows, lancez VMWare Horizon Client, ajoutez le serveur [vdi.epfl.ch](https://vdi.epfl.ch) et utilisez votre identifiant GASPARG;
- sur votre propre machine: en téléchargeant d'abord VMWare Horizon Client depuis <https://vdi.epfl.ch>, puis en procédant comme pour les salles CM1103/CM1112;

... puis choisissez la machine virtuelle IC-CO-IN-SC-INJ (et pas une autre)

Notez bien qu'à chaque logout, vos données sur l'ordinateur sont **effacées**. Seulement le contenu de votre dossier **myfiles**, visible sur votre bureau après l'ouverture de la machine virtuelle, est sauvegardé et réapparaît au prochain login.

Faites donc attention à toujours travailler dans votre dossier réseau!

Vous pouvez aussi accéder à **myfiles** en le montant comme dossier réseau sur votre propre machine. Si vous n'êtes pas sur le réseau de l'EPFL, vous devrez vous connecter au VPN d'abord. Plus d'info: <http://mynas.epfl.ch>; <http://studinfo.epfl.ch/core/index.asp?article=18>; <https://vpn.epfl.ch>.

### Configuration

Voici la procédure pour configurer le tout sur un poste de travail virtuel. **Ceci, sauf indication contraire, est à faire une seule fois lors de la première séance d'exercices:**

1. Loggez-vous sur une machine virtuelle (via une des machines de la salle d'exercice ou via votre propre machine par l'intermédiaire de VMware Horizon Client comme indiqué sur <https://vdi.epfl.ch>).
2. Lancez **Firefox** depuis la barre latérale, puis allez sur la page Moodle du cours et téléchargez (dans votre dossier **Downloads**, par défaut) le fichier de configuration **setup.sh**.
3. Ouvrez l'application **Terminal** (par exemple via le lanceur en bas de la barre latérale), puis tapez exactement ceci, ligne par ligne et en faisant bien attention aux espaces:  

```
cd Downloads
chmod +x setup.sh
./setup.sh
```
4. Observez la machine travailler pour vous. Cela prend un certain temps. À la fin de l'opération, s'il n'y a pas eu d'erreur, fermez la fenêtre du terminal puis lancez **Visual Studio Code** (aussi depuis le lanceur en bas à gauche). Choisissez ensuite **File** → **Open Workspace...**, puis naviguez vers votre fichier workspace, qui est ce fichier:  
`/Desktop/myfiles/ICC_Programmation/icc_prog.code-workspace`.
5. Dans la liste des fichiers de gauche, repérez et ouvrez **test.py**. Cliquez ensuite sur le petit bouton en forme de Play en haut à droite. Cela prend un peu de temps la première fois, mais devrait finalement vous afficher comme output  
**Welcome to Python!**  
Si ce n'est pas le cas, appelez l'enseignant ou un·e assistant·e.

Rappel: cette procédure est à effectuer uniquement la première fois que vous accédez à votre machine virtuelle. Les fois suivantes, vous ouvrez directement VS Code, puis faites **Open Workspace...** si nécessaire pour retrouver l'ensemble de vos fichiers de travail.

## Si vous utilisez votre propre machine:

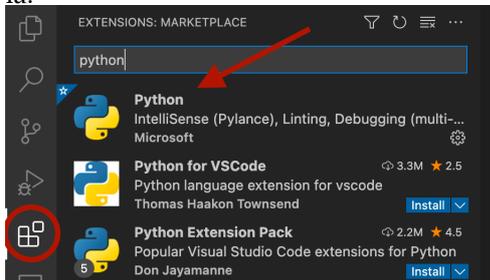
### Disponibilité

Il est de votre propre responsabilité de faire en sorte que votre machine de travail soit opérationnelle, à jour, connectée au wifi, etc. Les machines virtuelles sont toujours une solution de secours possible s'il vous manque votre machine un jour, mais c'est plus agréable de toujours travailler dans le même environnement de travail.

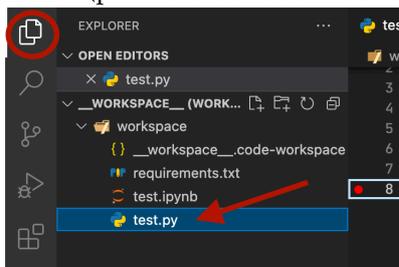
### Configuration

Attention, cette procédure a été testée avec plusieurs configurations d'ordinateurs, mais pas toutes celles que vous êtes susceptibles d'avoir. En cas de souci, contactez un·e assistant·e!

1. Téléchargez et installez Python 3.10 pour votre système d'exploitation.  
**macOS:** <https://www.python.org/downloads/macos/>  
(ou, si vous connaissez Homebrew, faites depuis le terminal: `brew install python3.10`)  
**Linux:** via votre package manager (demandez à un assistant·e si vous ne savez pas comment faire)  
**Windows:** <https://www.python.org/downloads/windows/>
2. Téléchargez et installez Visual Studio Code: <https://code.visualstudio.com/download>.
3. Ouvrez Visual Studio Code et sélectionnez, dans la barre de gauche, l'icône qui affiche les extensions. Faites une recherche avec «python» pour trouver l'extension Python de Microsoft. Installez-la.



4. Téléchargez depuis la page Moodle du cours le dossier compressé de base dans lequel nous allons travailler. Décompressez-le où vous souhaitez le ranger. Supprimez si nécessaire le fichier zip une fois décompressé.
5. Dans Visual Studio Code, assurez-vous que l'installation de l'extension du point 3 est terminée. À partir du menu **File** et **Open Workspace**, choisissez le fichier **icc\_prog.code-workspace** dans le dossier décompressé du point 4. Ouvrez ensuite le fichier **test.py** à partir de l'explorateur de fichier (première icône de la barre de gauche).



6. En bas à gauche de la fenêtre, cliquez sur **Select Python Interpreter**. Si un choix de workspaces vous est présenté, choisissez **Entire workspace**. Dans la liste suivante, choisissez l'entrée qui correspond à la version de Python 3.10 que vous venez d'installer ou qui était déjà présente sur votre machine.

7. Quittez Visual Studio Code et relancez-le. Si votre workspace n'est pas automatiquement rechargé, rouvrez-le comme à l'étape 5.
8. Si Visual Studio Code vous propose, via une alerte en bas à droite, d'installer **mypy** et/ou **pylint**, acceptez et effectuez l'installation.
9. Rouvrez le fichier **test.py**. Cliquez ensuite sur le petit bouton en forme de Play en haut à droite. Cela prend un peu de temps la première fois, mais devrait finalement vous afficher comme output **Welcome to Python!**  
Si ce n'est pas le cas, appelez l'enseignant ou un-e assistant-e.

## Exercice 2. Hello World

### Visual Studio Code

Visual Studio Code est l'application que nous utiliserons pour développer en Python. Tout ce que vous créez comme fichier dans VS Code sera enregistré dans le **workspace**, l'espace de travail — autrement dit, le dossier qui contient tous les fichiers Python que vous écrirez ainsi que les informations propre à VS Code sur votre workspace. Au démarrage de VS Code, votre première action sera toujours d'aller ouvrir votre workspace, représenté par le fichier **icc\_prog.code-workspace** dans le dossier dans lequel vous travaillerez. Vous réutiliserez le même workspace tout le semestre et, ainsi, conserverez votre code d'une semaine à l'autre.

Lorsque vous créez des fichiers Python pour les séries, habituez-vous à structurer votre code: par exemple, nommez vos fichiers en fonction de la série et du numéro de l'exercice (**s01e01.py**). Les fichiers Python doivent se terminer par l'extension **.py**.

- (a) Créez un nouveau fichier qui s'appelle **s01e01.py**. D'abord, stockez dans une nouvelle variable appelée **my\_name** une chaîne de caractères qui indique votre prénom. Écrivez ensuite une ligne qui affiche **Hello, <votre nom>!**, où **<votre nom>** est déterminé par la variable **my\_name**. Vous pouvez utiliser le modèle ci-dessous, où vous remplacez les points de suspension par du vrai code.

```
1 my_name = ...
2 print( ... )
```

- (b) Modifiez votre programme pour faire en sorte que la longueur de votre nom s'affiche en dessous de la ligne de salutations. Vous devez obtenir cette information en «demandant à Python» de déterminer la longueur du string en question, et non de lui faire afficher un nombre que vous prédéfinissez. Pour ce faire, consultez le cours pour savoir comment utiliser la fonction **len()**.

*Indice:* si vous avez une variable qui s'appelle **my\_variable** et qui contient une chaîne de caractères, l'expression **len(my\_variable)** vous renvoie une valeur de type **int** qui est la longueur du string **my\_variable**.

Testez ensuite votre code avec plusieurs valeurs différentes pour la variable **my\_name** pour vous convaincre que votre code fonctionne dans des circonstances différentes.

## Exercice 3. Manipulation de chaînes de caractères

En essayant le code, ou en faisant des recherches sur internet, trouvez ce que produisent comme valeurs les expressions **my\_name[1:3]** et **my\_name.upper()**. Modifiez ensuite votre programme de l'exercice 2 pour qu'il affiche votre nom:

- (a) tout en majuscules;
- (b) avec seulement la première lettre en majuscule;
- (c) avec un trait d'union inséré au milieu du string. Indiquez comment vous interprétez dans votre code «milieu du string» et vérifiez le bon fonctionnement de votre code pour des prénoms de longueurs paires et impaires.

## Exercice 4. Arithmétique des nombres entiers & interpolation dans les strings

Considérez ce programme:

```
1 distance = 10
2 speed = 3
3 duration = distance // speed
4
5 print(f"Distance à parcourir: {distance} km")
6 print(f"Vitesse: {speed} km/h")
7 print(f"Durée du trajet: {duration} h")
```

- (a) Que se passe-t-il si vous enlevez le **f** qui précède les strings donnés comme arguments à **print**? Quelle est donc l'utilité de ce **f**?
- (b) Qu'est-ce que ce programme est censé calculer? Que fait-il à la place?
- (c) Insérez un breakpoint sur la première ligne et exécutez ce programme ligne par ligne. Quel endroit pose problème? Qu'en déduisez-vous sur les opérateurs de division? Corrigez la ou les lignes concernées.
- (d) Faites en sorte que ce programme affiche la durée du trajet à la minute près en affichant ceci:

**Durée du trajet: 3 h 20 min.**

Vous aurez fort probablement besoin d'arrondir, de soustraire et de multiplier des valeurs. Référez-vous aux slides du cours et, si nécessaire, cherchez sur Internet les fonctions qui peuvent vous aider. Testez avec d'autres valeurs initiales pour vous assurer que votre code est correct.