Algorithmes et Pensée Computationnelle

Introduction à Python

1 Objectifs du document

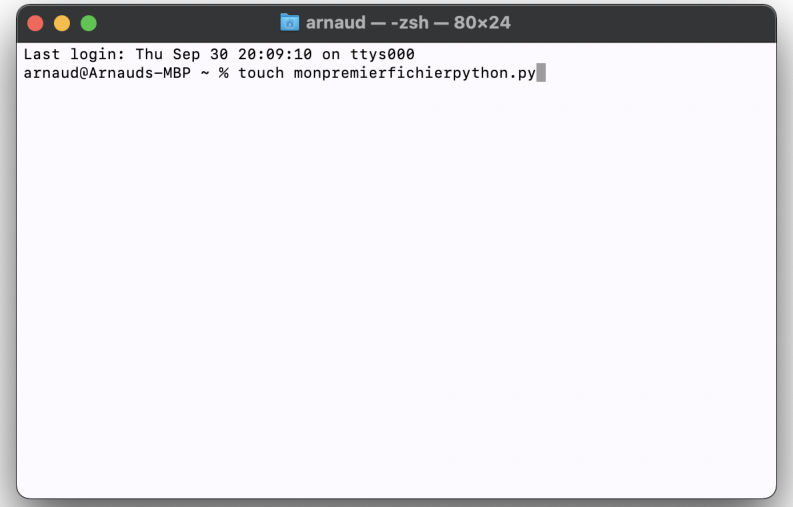
Ce document constitue un guide pour débuter à programmer en utilisant le langage Python. Les objectifs de ce guide sont les suivants :

- Comprendre comment créer, éditer et lancer un script Python.
- Découvrir l'environnement de travail utilisé pour développer en Python.
- Se familiariser avec quelques notions de base pour commencer à programmer en Python.

2 Introduction

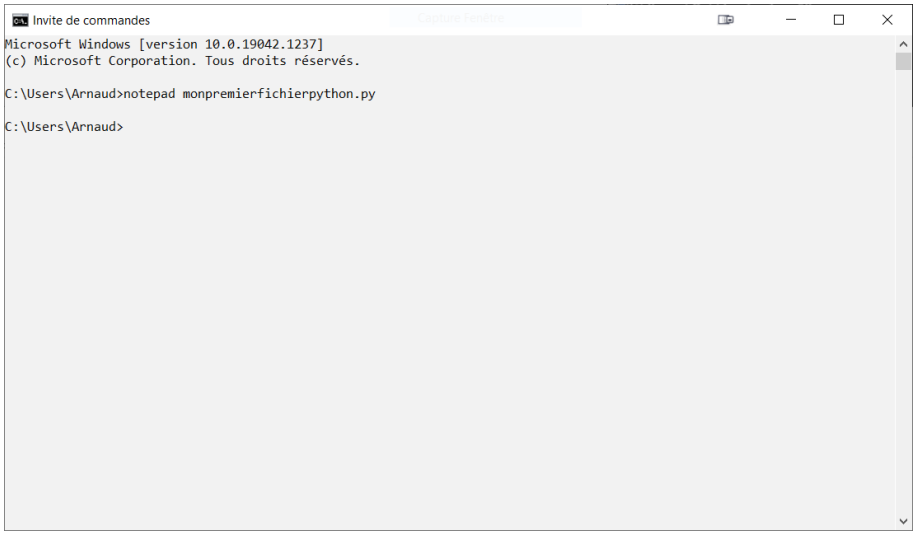
2.1 Création d'un script Python sur Mac et Linux

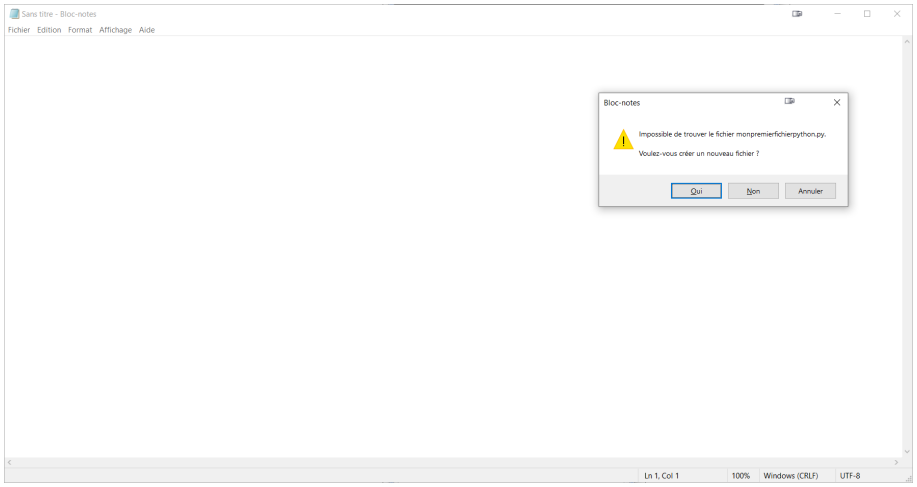
Pour commencer à programmer en Java, il est nécessaire de créer un fichier ayant une extension **.py**. Pour ce faire, il suffit de créer un simple fichier python vide. La manière la plus simple est de passer par le terminal (ou invite de commande). Une fois ouvert, entrer la commande **touch hello.py** ou **nano hello.py** pour créer un fichier vide nommé **hello.py**.



2.2 Création d'un script Python sur Windows

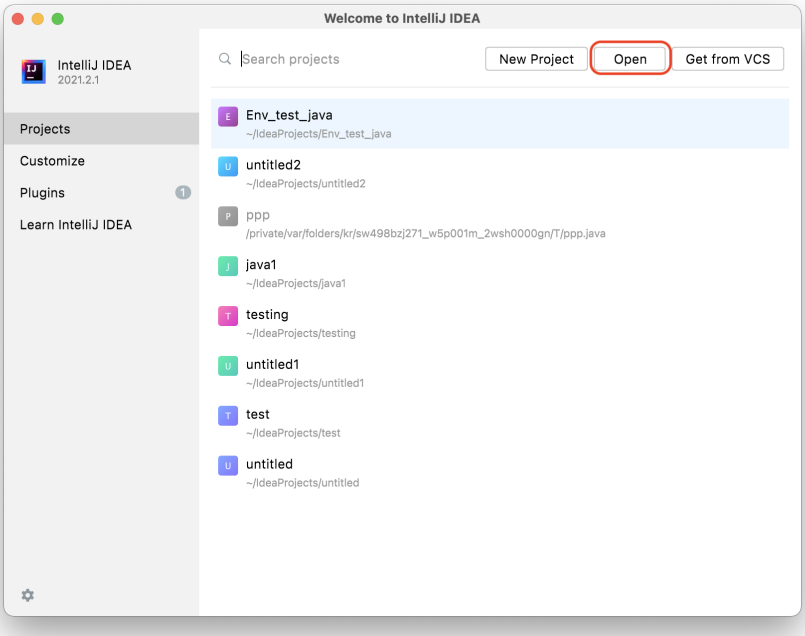
Pour créer un fichier Python sur Windows, il suffit simplement d'entrer la commande **notepad hello.py** dans le bloc-note. Vous pouvez alors simplement l'éditer et l'enregistrer.



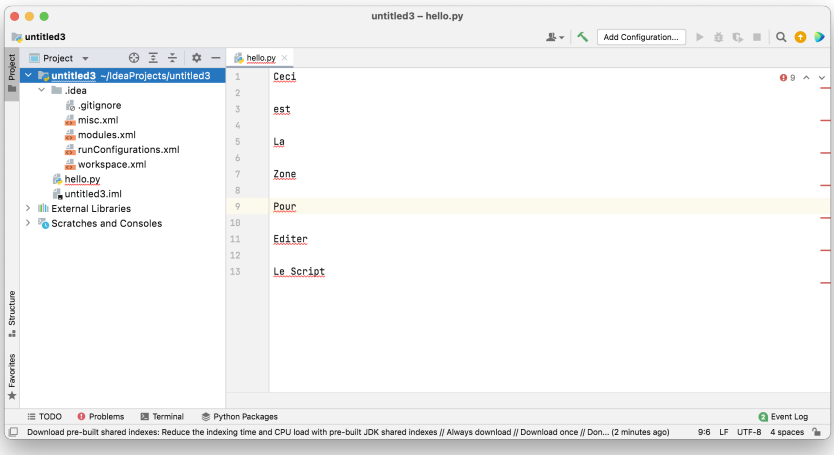


2.3 IntelliJ IDEA

Dans le cours, il vous a été demandé d'installer Intellij IDEA, un des l'IDE les plus populaire de ces dernières années. Un IDE (Integrated Developement Environnement) est un programme qui combine différents outils de développement et qui facilite grandement le travail d'un programmeur. La première chose à faire avec le programme est ouvrir et exécuter un script python. Pour ce faire, lancez le programme IntelliJ IDEA et ouvrez le fichier **hello.py**.



La fenêtre qui vient de s'ouvrir est similaire à un éditeur de texte basique. C'est ici que le code peut être entré et modifié. Notez qu'il est aussi possible de créer directement un fichier depuis votre IDE.



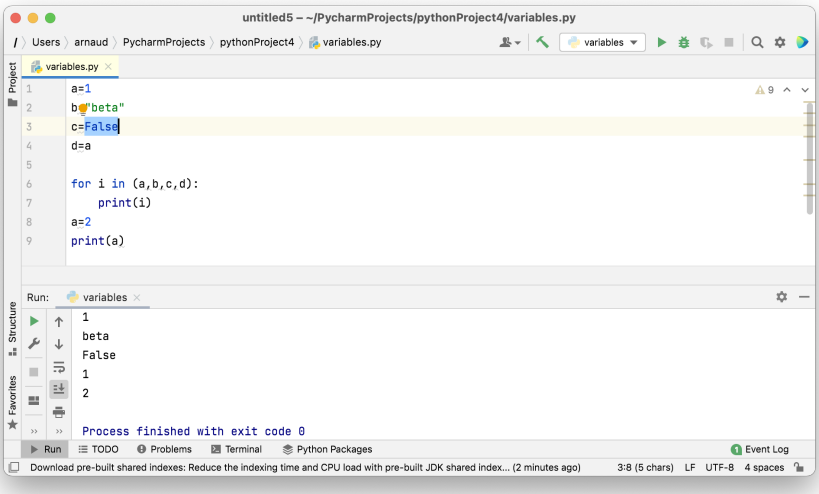
2.4 Création de variables

Les variables dans les langages de programmation sont similaires à des noms donnés à une valeur précise. Pour assigner une valeur à une variable en Python, il suffit de respecter la forme suivante **variable=valeur**.

Conseil

- En Python, le type de la variable est défini de façon automatique.
- On peut assigner n'importe quelle suite de caractères non-reservée en tant que variable.
- Il est aussi possible d'assigner des chaînes de caractères (strings en anglais) ou encore des valeurs booléennes à des variables.
- En réassignant une nouvelle valeur à une variable déjà définie, la valeur de la variable va être écrasée et remplacée par la nouvelle valeur.
- Plusieurs variables peuvent avoir la même valeur.

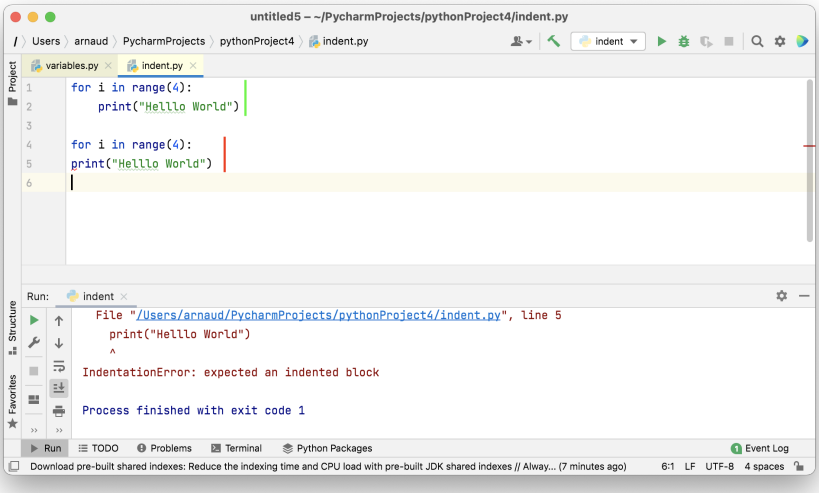
Si vous ouvrez et exécutez le script Python **variables.py** (se trouvant sur Moodle, dans le dossier Ressources) qui contient quelques exemples d'attribution de variables, vous pourrez observer comment vous pouvez créer différentes variables mais aussi comment le programme les traite.



2.5 L'indentation

Python est un langage de programmation sensible aux erreurs d'indentation. Une indentation correspond à une tabulation ou un espace. Il sera donc important de bien comprendre comment celles-ci fonctionnent. Prenons par exemple une simple boucle **for** qui sera traitée un peu plus loin dans ce guide.

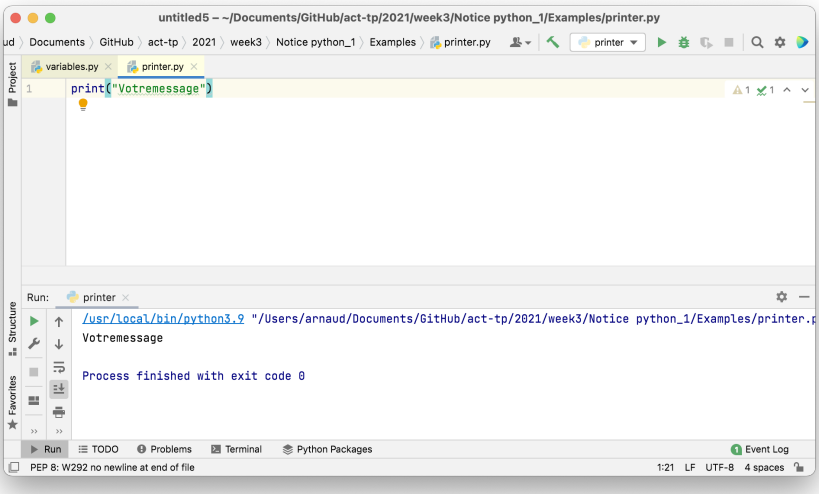
Si vous ouvrez et exécutez les deux scripts Python **indent1.py** et **indent2.py** en allant sur le barre de menu (tout en haut) et en cliquant sur **Run** > **Run `hello'**, vous vous rendrez compte que l'un des deux produit une erreur alors que l'autre fonctionne correctement. En effet, dans un des scripts, la console produit une erreur d'indentation. Après une boucle **for**, il est nécessaire de bien respecter l'indentation pour définir son utilité. L'indentation est importante dans la syntaxe d'un script Python. Une erreur d'indentation peut changer le fonctionnement d'un script ou tout simplement empêcher celui-ci de s'exécuter.



2.6 Les fonctions

Dans les langages de programmation, on retrouve un très grand nombre de fonctions. Ces fonctions sont des blocs de code qui, lorsqu'ils sont invoqués avec certains paramètres, effectuent certaines actions. Une des

fonctions basique et plutôt importante en Python est la fonction **print()**. Cette fonction permet d'afficher à l'écran le contenu de la parenthèse.
Si vous entrez la ligne de code **print("Votremessage")** et que vous exécutez le script, IntelliJ IDEA va vous afficher une ligne de texte. Le message qui apparaît est celui que vous avez entré entre guillemets (à la place de "Votremessage"). C'est le but d'une fonction **print()**.



En plus des fonctions incluses dans les librairies Python, il est possible de créer des fonctions complètement personnalisées (généralement plusieurs fonctions sont réunies en une seule) au moyen de la fonction **def nomdelafonction():** et de l'utiliser à n'importe quel moment en l'invoquant avec **nomdelafonction()**.
Si vous ouvrez et exécutez le script python **Arbre.py** (se trouvant sur Moodle, dans le dossier Exercices (version hors ligne)/Code), vous pourrez voir l'exemple de la fonction personnalisée **def build_tree(t, branch_length, shorten_by, angle)** qui permet de dessiner un arbre à l'aide d'un algorithme. Nous verrons plus tard dans ce cours comment fonctionnent ces algorithmes.