

Introduction et configuration du jour 1

Bienvenue dans votre voyage dans le monde de la programmation Python avec ce challenge de Python Togo. Au cours des 30 prochains jours, vous explorerez l'un des langages de programmation les plus populaires et les plus adaptés aux débutants utilisés dans des secteurs comme le développement Web, la science des données, l'automatisation et l'intelligence artificielle. Python est un langage de programmation de haut niveau, interprété et à usage général. Connu pour sa syntaxe simple et lisible, Python est largement utilisé par les débutants et les professionnels. Il a été créé par Guido Van Rossum et publié pour la première fois en 1991 dans le but de rendre le code plus compréhensible et plus accessible. Depuis lors, il est devenu l'un des outils les plus puissants du développement de logiciels modernes. Ce cours est structuré pour vous aider à apprendre Python étape par étape - des bases mêmes aux applications réelles. Chaque jour présente un sujet ciblé ainsi que des explications claires, des exemples pratiques et des exercices pratiques pour vous aider à appliquer ce que vous apprenez immédiatement. Que vous parliez de zéro ou que vous reveniez au codage après une pause, ce cours est conçu pour renforcer vos compétences en confiance et en résolution de problèmes avec Python. Vous comprendrez comment rédiger du code propre, réfléchir logiquement et commencer à créer des programmes utiles par vous-même. Aucune expérience préalable n'est requise - simplement votre curiosité et votre engagement à apprendre quelque chose de nouveau chaque jour. À la fin de ce défi, vous comprendrez non seulement le fonctionnement de Python, mais vous aurez également les bases d'explorer des domaines plus avancés comme l'automatisation, les scripts ou l'analyse des données.

Commençons.

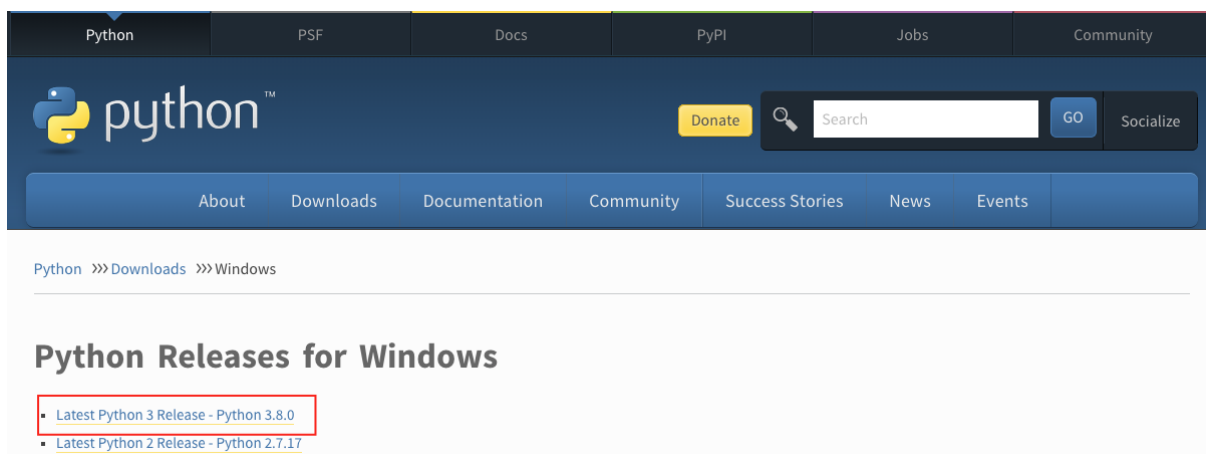
Pourquoi Python

Il s'agit d'un langage de programmation qui est très proche du langage humain et à cause de cela, il est facile à apprendre et à utiliser. Python est utilisé par diverses industries et entreprises (y compris Google). Il a été utilisé pour développer des applications Web, des applications de bureau, une administration système et des bibliothèques d'apprentissage automatique. Python est un langage très adopté dans la communauté des sciences et de l'apprentissage automatique des données. J'espère que cela suffit pour vous convaincre de commencer à apprendre Python. Python mange le monde et vous le tue avant qu'il ne vous mange.

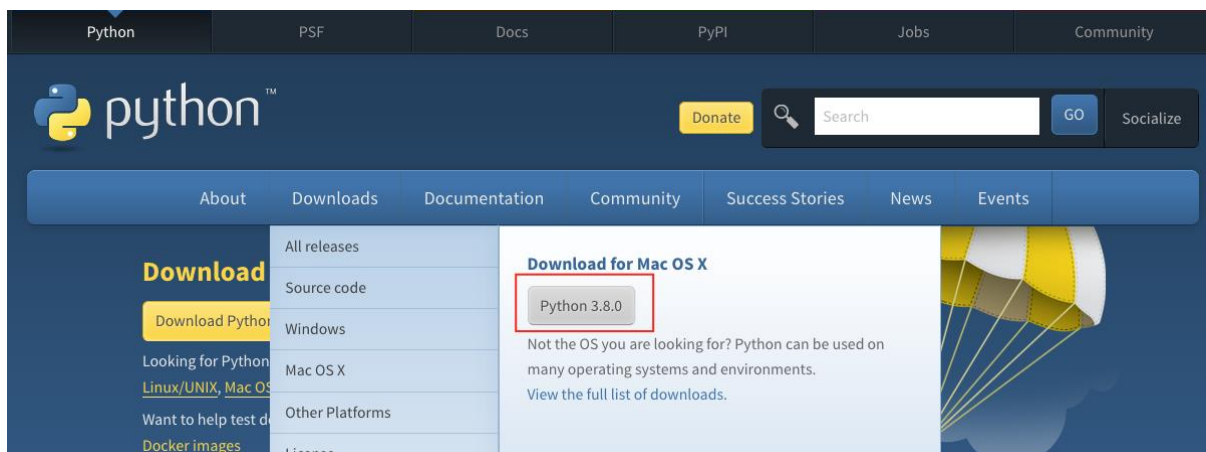
Configuration de l'environnement

Installation de Python

Pour exécuter un script Python, vous devez installer Python. Téléchargeons Python. Si vous êtes un utilisateur Windows. Cliquez sur le bouton entouré en rouge.



Si vous êtes un utilisateur macOS. Cliquez sur le bouton entouré en rouge.



Pour vérifier si Python est installé, écrivez la commande suivante sur le terminal de votre périphérique.

Python - Version



```
asabeneh ~ -bash — 80x24
Last login: Tue Nov 19 15:45:18 on ttys002
asabeneh@Asabeneh:~$ python --version
Python 3.7.5
asabeneh@Asabeneh:~$
```

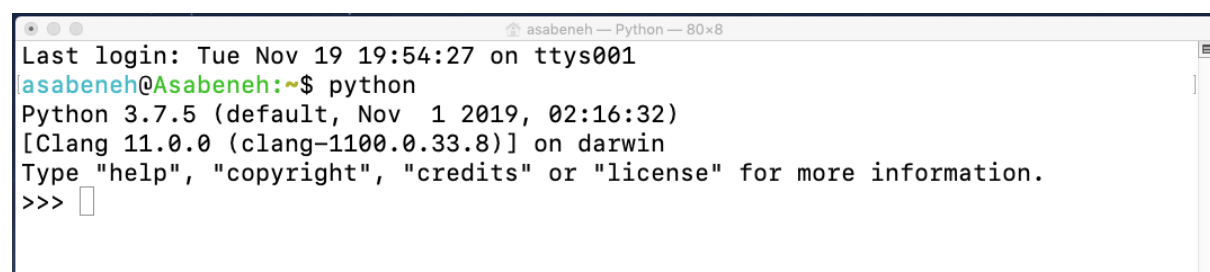
Comme vous pouvez le voir sur le terminal, j'utilise la version *Python 3.7.5* en ce moment. Votre version de Python peut être différente de la mienne, mais elle devrait être 3,6 ou plus. Si vous gêchez pour voir la version Python, bravo. Python a été installé sur votre machine. Continuez à la section suivante.

Coquille de python

Python est un langage de script interprété, il n'a donc pas besoin d'être compilé. Cela signifie qu'il exécute le code ligne par ligne. Python est livré avec un *Python Shell (Python Interactive Shell)*. Il est utilisé pour exécuter une seule commande python et obtenir le résultat.

Python Shell attend le code Python de l'utilisateur. Lorsque vous entrez le code, il interprète le code et affiche le résultat dans la ligne suivante. Ouvrez votre terminal ou votre invite de commande (CMD) et écrivez:

python



```
asabeneh ~ Python — 80x8
Last login: Tue Nov 19 19:54:27 on ttys001
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Le shell interactif Python est ouvert et il vous attend pour écrire du code Python (script Python). Vous écrirez votre script Python à côté de ce symbole `>>>`, puis cliquez sur Entrer. Écrivez notre tout premier script sur le shell de script Python.

```
asabeneh — Python — 80x10
Last login: Tue Nov 19 20:05:55 on ttys001
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 3
5
>>> █
```

Bravo, vous avez écrit votre premier script Python sur Python Interactive Shell. Comment fermer le shell interactif Python? Pour fermer le shell, à côté de ce symbole >> écrivez l'exit () et appuyez sur Entrée.

```
asabeneh — Python — 80x10
Last login: Tue Nov 19 20:05:55 on ttys001
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 3
5
>>> exit() █
```

Maintenant, vous savez comment ouvrir le shell interactif Python et comment en sortir.

Python vous donnera des résultats si vous écrivez des scripts que Python comprend, sinon il renvoie des erreurs. Faisons une erreur délibérée et voyons ce que Python reviendra.

```
asabeneh — Python — 80x12
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 3
5
>>> 2 x 3
File "<stdin>", line 1
  2 x 3
    ^
SyntaxError: invalid syntax
>>> █
```

Comme vous pouvez le voir à partir de l'erreur renvoyée, Python est si intelligent qu'il connaît l'erreur que nous avons commise et qui était *Syntax Error: invalid syntax*. L'utilisation de X comme multiplication dans Python est une erreur de syntaxe car (x) n'est pas une syntaxe valide dans Python. Au lieu de (x), nous utilisons l'astérisque (*) pour la multiplication. L'erreur renvoyée montre clairement quoi corriger.

Le processus d'identification et de suppression des erreurs d'un programme est appelé *debugging*. Laissez-nous le déboguer en mettant * à la place de x.

```
asabeneh — Python — 80x15
Last login: Tue Nov 19 20:05:55 on ttys001
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 2 + 3
5
>>> 2 x 3
      File "<stdin>", line 1
        2 x 3
          ^
SyntaxError: invalid syntax
>>> 2 * 3
6
>>> █
```

Notre bogue a été corrigé, le code s'est déroulé et nous avons obtenu un résultat que nous attendions. En tant que programmeur, vous verrez ce type d'erreurs quotidiennement. Il est bon de savoir comment déboguer. Pour être doué pour le débogage, vous devez comprendre quel type d'erreurs vous êtes confrontés. Certaines des erreurs Python que vous pouvez rencontrer

sont *SyntaxError*, *IndexError*, *NameError*, *ModuleNotFoundError*, *KeyError*, *ImportError*, *AttributeError*, *TypeError*, *ValueError*, *ZeroDivisionError* etc. Nous verrons plus sur différents types d'erreur de python dans les sections ultérieures.

Passons plus à l'utilisation de Shell interactif Python. Accédez à votre terminal ou à votre invite de commande et écrivez le mot Python.

```
asabeneh — Python — 80x8
Last login: Tue Nov 19 19:54:27 on ttys001
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

La coque interactive Python est ouverte. Faisons quelques opérations mathématiques de base (addition, soustraction, multiplication, division, module, exponentielle).

Faisons d'abord quelques mathématiques avant d'écrire un code Python:

- $2 + 3$ est 5 • $3 - 2$ est 1 • $3 *$
- 2 est 6 • $3/2$ est 1,5 • $3 ** 2$ e
- st le même que $3 * 3$

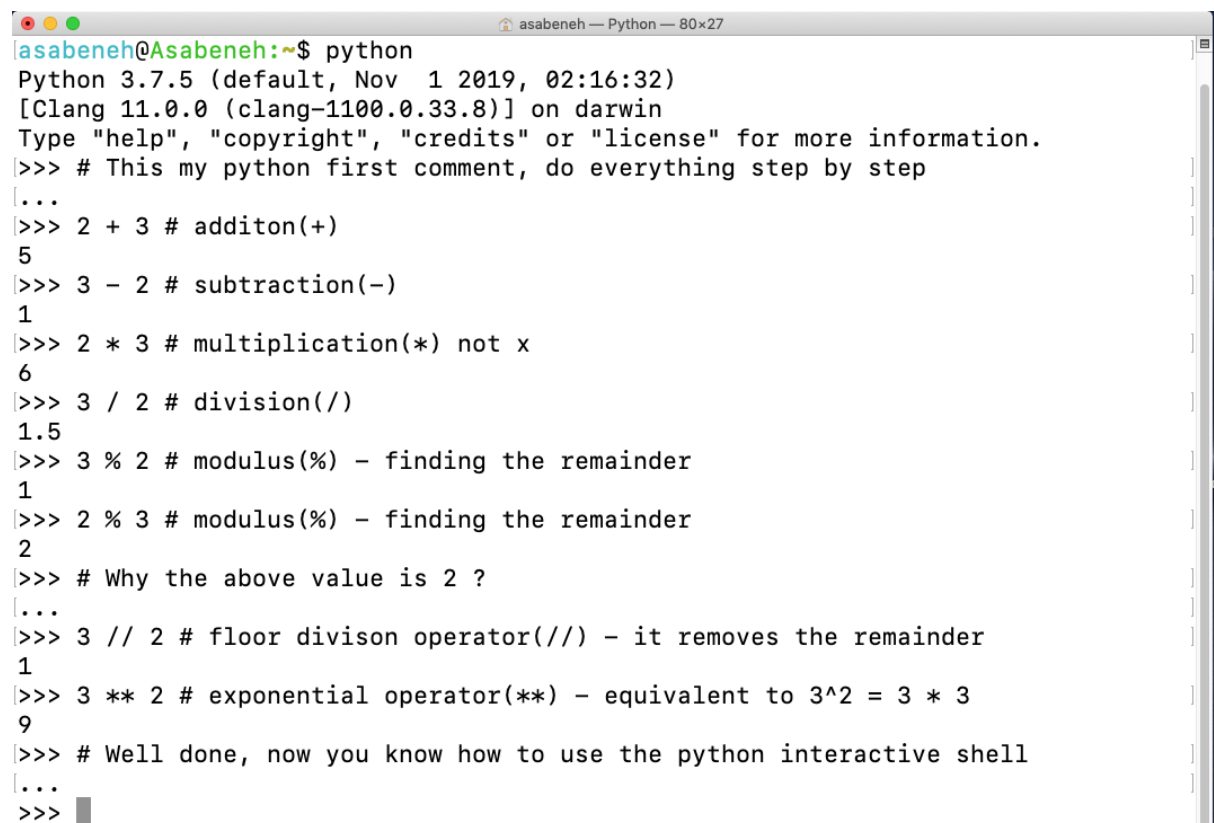
Dans Python, nous avons les opérations supplémentaires suivantes:

- $3 \% 2 = 1$ => qui signifie trouver le reste • $3 // 2 = 1$
- => qui signifie supprimer le reste

Changeons les expressions mathématiques ci-dessus en code python. La coquille Python a été ouverte et écrivons un commentaire au tout début de la coquille.

Une *comment* fait partie du code qui n'est pas exécuté par Python. Nous pouvons donc laisser un texte dans notre code pour rendre notre code plus lisible. Python n'exécute pas la partie commentaire. Un commentaire dans Python commence par le symbole de hachage (#). C'est ainsi que vous écrivez un commentaire dans Python

```
# Le commentaire commence par Hash # Ceci est un commentaire Python, car il commence par un (#) symbole
```

A screenshot of a terminal window titled 'asabeneh — Python — 80x27'. The prompt is 'asabeneh@Asabeneh:~\$'. The user has entered 'python', which has started the Python 3.7.5 interpreter. The prompt is now '>>>'. The user enters a series of arithmetic operations, each followed by a comment: '>>> # This my python first comment, do everything step by step', '>>> ...', '>>> 2 + 3 # additon(+)', '>>> 5', '>>> 3 - 2 # subtraction(-)', '>>> 1', '>>> 2 * 3 # multiplication(*) not x', '>>> 6', '>>> 3 / 2 # division(/)', '>>> 1.5', '>>> 3 % 2 # modulus(%) - finding the remainder', '>>> 1', '>>> 2 % 3 # modulus(%) - finding the remainder', '>>> 2', '>>> # Why the above value is 2 ?', '>>> ...', '>>> 3 // 2 # floor divison operator(//) - it removes the remainder', '>>> 1', '>>> 3 ** 2 # exponential operator(**) - equivalent to 3^2 = 3 * 3', '>>> 9', '>>> # Well done, now you know how to use the python interactive shell', '>>> ...', and finally '>>>' with a cursor. The output of each operation is displayed on the line immediately following the input.

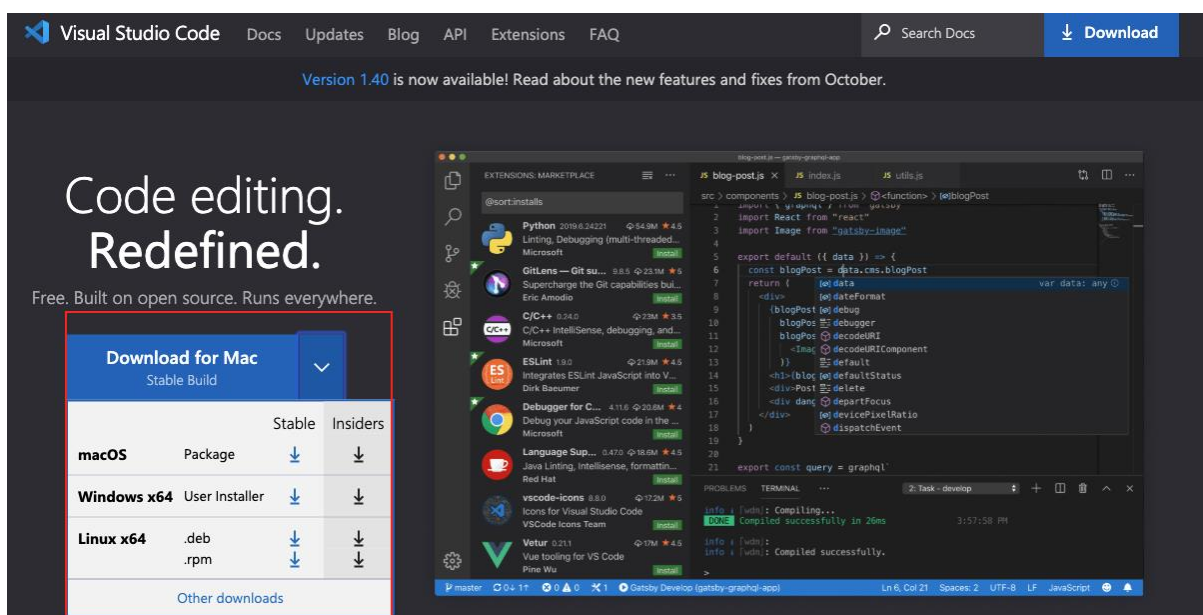
```
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> # This my python first comment, do everything step by step
>>> ...
>>> 2 + 3 # additon(+)
5
>>> 3 - 2 # subtraction(-)
1
>>> 2 * 3 # multiplication(*) not x
6
>>> 3 / 2 # division(/)
1.5
>>> 3 % 2 # modulus(%) - finding the remainder
1
>>> 2 % 3 # modulus(%) - finding the remainder
2
>>> # Why the above value is 2 ?
>>> ...
>>> 3 // 2 # floor divison operator(//) - it removes the remainder
1
>>> 3 ** 2 # exponential operator(**) - equivalent to 3^2 = 3 * 3
9
>>> # Well done, now you know how to use the python interactive shell
>>> ...
>>>
```

Avant de passer à la section suivante, pratiquons davantage sur le shell interactif Python. Fermez le shell ouvert en écrivant *exit()* sur le shell et ouvrez-le à nouveau et pratiquons comment écrire du texte sur le shell Python.

```
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> # Let's practice how to write text on python shell: we use single or double quote to make a string. Any text in a quote we call it string
...
>>> 'Asabeneh' # To write my name as string
'Asabeneh'
>>> "Asabeneh" # To write my name as string - now in double quote
'Asabeneh'
>>> # A string could be a single character text or a page
...
>>> 'I love teaching. And thank you so much for joining this course'
'I love teaching. And thank you so much for joining this course'
>>> # If the text is too long we use triple quote to make a string('''''')
...
>>> '''We use triple quote if the text is more than one line'''
'We use triple quote if the text is more than one line'
>>> # Now you knew string and how to use the python shell
...
>>> # Let's continue to the next section and install text editor code editor
...
>>> exit()
asabeneh@Asabeneh:~$
```

Installation du code Visual Studio

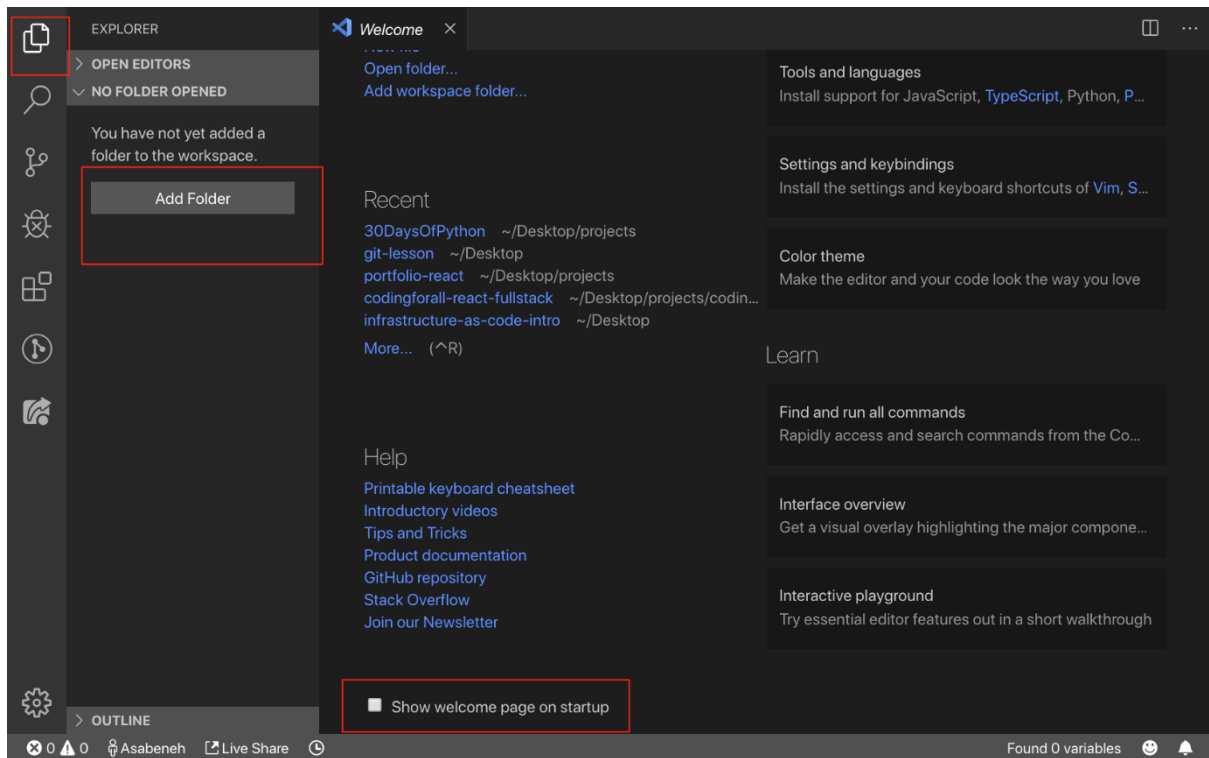
Le shell interactif Python est bon d'essayer de tester de petits codes de script, mais ce ne sera pas pour un grand projet. Dans un environnement de travail réel, les développeurs utilisent différents éditeurs de code pour écrire des codes. Au cours de ce défi de programmation de 30 jours Python, nous utiliserons le code Visual Studio. Visual Studio Code est un éditeur de texte open source très populaire. Je suis un fan de VScode et je recommanderais de télécharger Visual Studio Code, mais si vous êtes en faveur d'autres éditeurs, n'hésitez pas à suivre ce que vous avez.



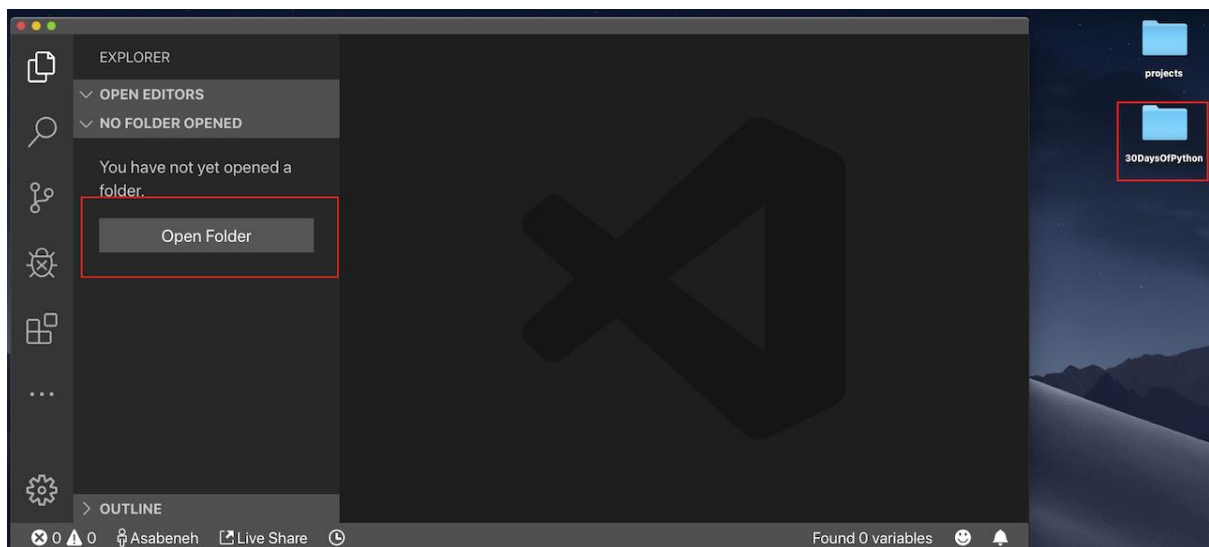
Si vous avez installé Visual Studio Code, voyons comment l'utiliser. Si vous préférez une vidéo, vous pouvez suivre ce code [Visual Studio pour Python Video Tutorial](#)

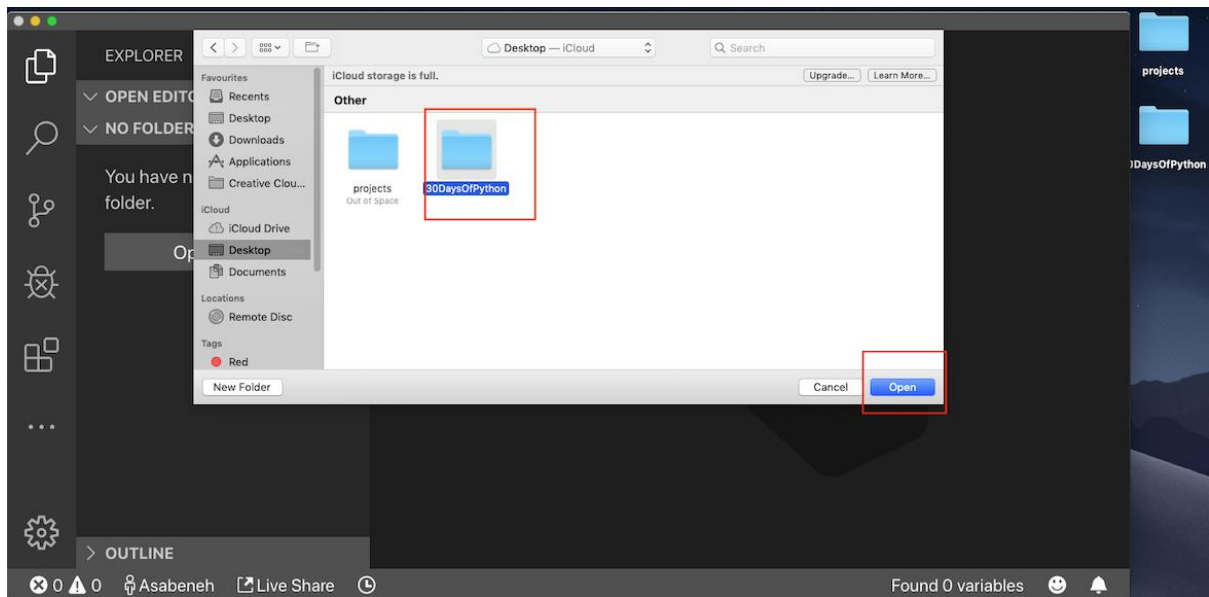
Comment utiliser le code Visual Studio

Ouvrez le code Visual Studio en double-cliquant sur l'icône Visual Studio. Lorsque vous l'ouvrez, vous obtiendrez ce type d'interface. Essayez d'interagir avec les icônes étiquetées.

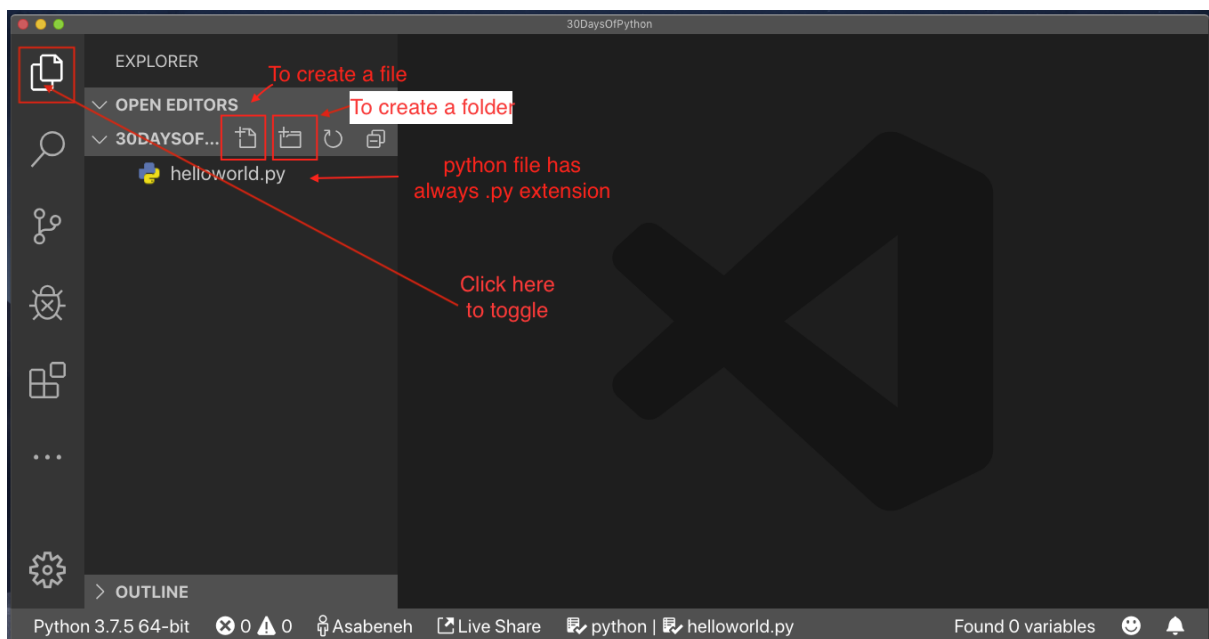


Créez un dossier nommé 30daysofpython sur votre bureau. Puis ouvrez-le à l'aide du code Visual Studio.

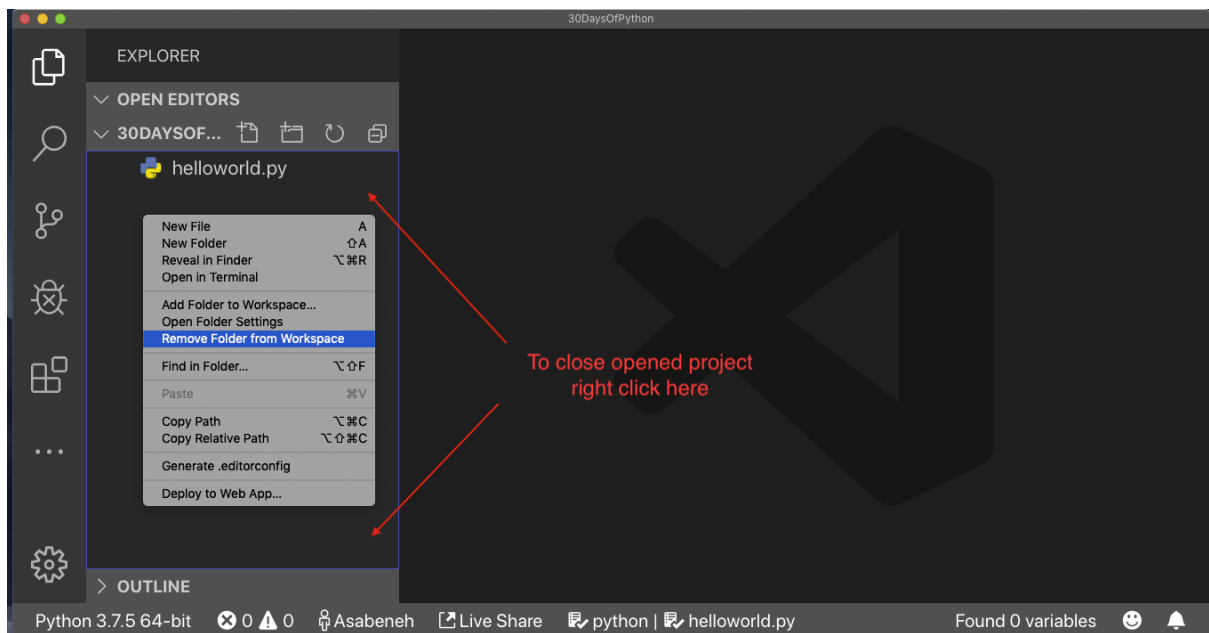




Après l'ouvrir, vous verrez des raccourcis pour la création de fichiers et de dossiers à l'intérieur du répertoire du projet de 30 jours. Comme vous pouvez le voir ci-dessous, j'ai créé le tout premier fichier, helloworld.py. Vous pouvez faire de même.



Après une longue journée de codage, vous souhaitez fermer votre éditeur de code, non? C'est ainsi que vous fermerez le projet ouvert.



Félicitations, vous avez fini de mettre en place l'environnement de développement. Commençons par coder.

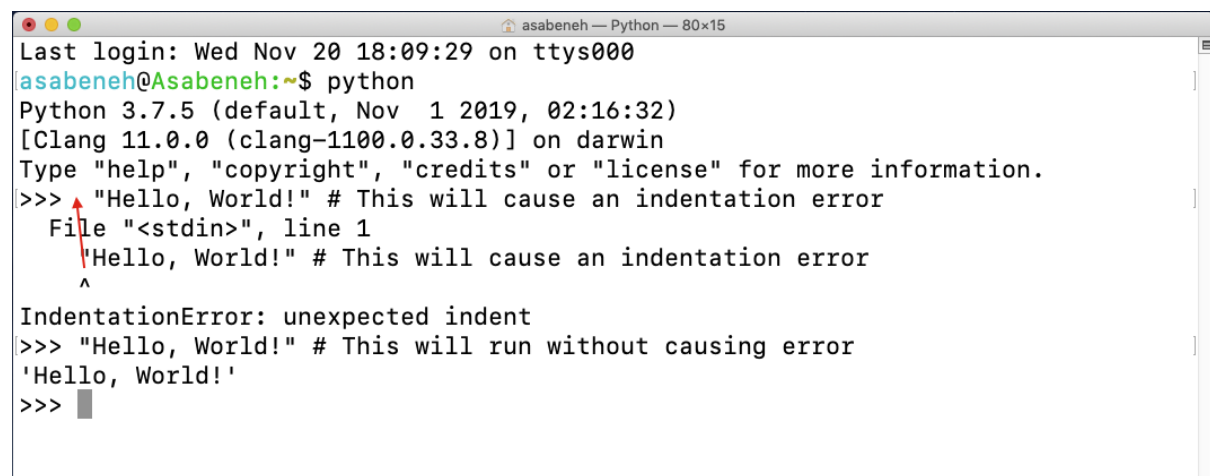
Python de base

Syntaxe python

Un script Python peut être écrit dans Python Interactive Shell ou dans l'éditeur de code. Un fichier Python a une extension .py.

Indentation python

Une indentation est un espace blanc dans un texte. L'indentation dans de nombreuses langues est utilisée pour augmenter la lisibilité du code; Cependant, Python utilise l'indentation pour créer des blocs de code. Dans d'autres langages de programmation, des supports bouclés sont utilisés pour créer des blocs de code au lieu de l'indentation. L'un des bogues communs lors de l'écriture de code Python est une indentation incorrecte.

A screenshot of a terminal window titled 'asabeneh — Python — 80x15'. The terminal shows the following text:

```
Last login: Wed Nov 20 18:09:29 on ttys000
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> "Hello, World!" # This will cause an indentation error
      File "<stdin>", line 1
        "Hello, World!" # This will cause an indentation error
        ^
IndentationError: unexpected indent
>>> "Hello, World!" # This will run without causing error
'Hello, World!'
>>> █
```

Commentaires

Les commentaires jouent un rôle crucial dans l'amélioration de la lisibilité du code et permettent aux développeurs de laisser des notes dans leur code. Dans Python, tout texte précédé d'un symbole de hachage (#) est considéré comme un commentaire et n'est pas exécuté lorsque le code s'exécute.

Exemple: commentaire à ligne unique

```
# This is the first comment
# This is the second comment
# Python is eating the world
```

Exemple: commentaire multiligne

La triple devis peut être utilisée pour des commentaires multilignes s'il n'est pas affecté à une variable

```
""" "Ce comment comment le commentaire multiline multil
ine prend plusieurs lignes.
Python mange le monde
""" "
```

Types de données

Dans Python, il existe plusieurs types de types de données. Commençons par les plus courants. Différents types de données seront couverts en détail dans d'autres sections. Pour le moment, passons simplement les différents types de données et nous familiariser avec eux. Vous n'avez plus besoin de compréhension maintenant.

Nombre

- Entier: entier (négatif, zéro et positif) Exemple de nombres: ... -3, -2, -1, 0, 1, 2, 3 ...
- float: nombre décimal Exemple ... -3,5, -2.25, -1.0, 0.0, 1.1, 2.2, 3.5 ...
- Complexe Exemple $1 + j$, $2 + 4j$

Chaîne

Une collection d'un ou plusieurs personnages sous une seule ou double citation. Si une chaîne est plus d'une phrase, nous utilisons une triple devise.

Exemple:

```
`` Asabeneh " Finlande " Python " j'aime enseigner " J'espère que vous profitez du premier jour du défi 30 jours sur le défi "
```

Booléens

Un type de données booléen est soit une valeur vraie ou fausse. T et F doivent être toujours en majuscules.

Exemple:

```
Vrai # La lumière est-elle allumée? S'il est allumé, alors la valeur est vraie fausse # est la lumière allumée? S'il est désactivé, alors la valeur est fausse
```

Liste

Python List est une collection commandée qui permet de stocker différents éléments de type de données. Une liste est similaire à un tableau en JavaScript.

Exemple:

```
[0, 1, 2, 3, 4, 5] # Tous sont les mêmes types de données - une liste de nombres ['banana', 'orange', 'mango', 'avocado'] # tous les mêmes types de données - une liste de chaînes (fruits) ['Finland', `` Estonia `` , `` Sweden `` , `` Norvège `` ] # Tous les mêmes types de données - une liste de coordonnées (pays) (pays)
```

```
['Banane', 10, false, 9.81] # différents types de données dans la liste - chaîne, entier, booléen et float
```

Dictionnaire

Un objet de dictionnaire Python est une collection de données non ordonnée dans un format de paire clé.

Exemple:

```
{'First_name': 'Asabeneh', 'last_name': 'encoreayeh', 'country': 'Finlande', 'Age': 250, 'is_married': true, 'compétences': ['js', 'react', 'node', 'python']}
```

Tuple

Un tuple est une collection ordonnée de différents types de données comme la liste, mais les tuples ne peuvent pas être modifiés une fois qu'ils sont créés. Ils sont immuables.

Exemple:

```
('Asabeneh', 'Pawel', 'Brook', 'Abraham', 'lidiya') # names ('Earth', 'Jupiter', 'neptune', 'mars', 'Vénus', 'Saturn', 'Uranus', 'Mercury') # planètes
```

Ensemble

Un ensemble est une collection de types de données similaires à la liste et au tuple. Contrairement à la liste et au tuple, Set n'est pas une collection commandée d'articles. Comme dans les mathématiques, placer dans les magasins Python uniquement des éléments uniques. Dans les sections ultérieures, nous irons en détail sur chaque type de données Python.

Exemple:

```
{2, 4, 3, 5} {3.14, 9.81, 2,7} # L'ordre n'est pas important dans l'ensemble
```

Vérification des types de données

Pour vérifier le type de données de certaines données / variable, nous utilisons la fonction de type. Dans le terminal suivant, vous verrez différents types de données Python:

```
asabeneh — Python — 80x24
Last login: Wed Nov 20 00:49:23 on ttys001
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov 1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> type(10)
<class 'int'>
>>> type(3.14)
<class 'float'>
>>> type(1 + 3j)
<class 'complex'>
>>> type('Asabeneh')
<class 'str'>
>>> type([1, 2, 3])
<class 'list'>
>>> type({'name': 'Asab'})
<class 'dict'>
>>> type((2, 3, 4, 5))
<class 'tuple'>
>>> type({9.8, 3.14, 2.7})
<class 'set'>
>>> exit()
```

Fichier python

Ouvrez d'abord votre dossier de projet, 30DaysOfpython. Si vous n'avez pas ce dossier, créez un nom de dossier appelé 30daysofpython. À l'intérieur de ce dossier, créez un fichier appelé helloworld.py. Maintenant, faisons ce que nous avons fait sur Python Interactive Shell en utilisant le code Visual Studio.

Le shell interactif Python était imprimé sans utiliser d'impression, mais sur Visual Studio Code pour voir notre résultat, nous devons utiliser une fonction intégrée `_print()`. La fonction intégrée `print()` prend un ou plusieurs arguments comme suit `print('argument1', 'argument2', 'argument3')`. Voir les exemples ci-dessous.

Exemple:

Le nom du fichier est helloworld.py

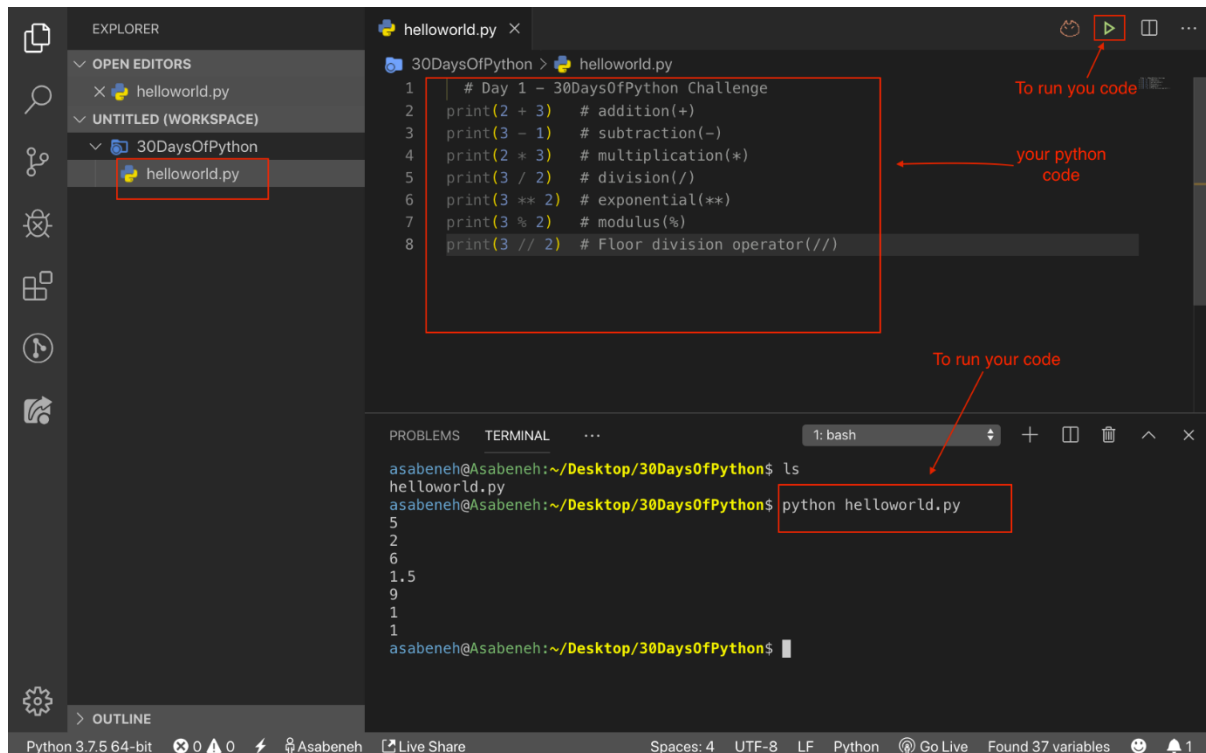
```
# Day 1 - 30DaysOfPython Challenge

print(2 + 3)           # addition(+)
print(3 - 1)           # subtraction(-)
print(2 * 3)           # multiplication(*)
print(3 / 2)           # division(/)
print(3 ** 2)          # exponential(**)
print(3 % 2)           # modulus(%)
print(3 // 2)          # Floor division operator(//)

# Vérification des types de données
```

```
print(type(10)) # int print(type(3.14)) # float print(type(1 + 3j)) #  
complexe numéro imprime (type('asabeneh')) # string print(type([1,  
2, 3])) # list print(type({'name': 'asabeneh'})) # dictionary imprimer  
(({9.8, 3.14, 2.7})) imprimer (type((9.8, 3.14, 2.7))) # tuple
```

Pour exécuter le fichier Python, vérifiez l'image ci-dessous. Vous pouvez exécuter le fichier Python soit en exécutant le bouton vert sur le code Visual Studio ou en tapant `python helloworld.py` dans le terminal.



Vous êtes incroyable. Vous venez de terminer le défi du jour 1 et vous êtes en route pour la grandeur. Faites maintenant quelques exercices pour votre cerveau et vos muscles.

Exercices - Jour 1

Exercice: niveau 1

1. Vérifiez la version Python que vous utilisez
2. Ouvrez le shell interactif Python et effectuez les opérations suivantes. Les opérandes sont 3 et 4.

o Addition (+) o Soustraction (-) o Multiplication (*) o Module (%) o Division (/) o Exponentielle (**) o Floor Division Operator (//)

3. Écrivez des chaînes sur la coque interactive Python. Les cordes sont les suivantes:

O votre nom o votre nom de famille
o votre pays o je profite de 30 jours
de python

4. Vérifiez les types de données des données suivantes:

O 10 O 9,8 O 3.14 O 4 - 4J O ['As
abeneh', 'Python', 'Finlande'] o Votre nom o votre nom de famille o votre pays

Exercice: niveau 2

1. Créez un dossier nommé Day_1 à l'intérieur du dossier 30daysofpython. À l'intérieur du dossier Day_1, créez un fichier python helloworld.py et répétez les questions 1, 2, 3 et 4. N'oubliez pas d'utiliser `print()` lorsque vous travaillez sur un fichier python. Accédez au répertoire où vous avez enregistré votre fichier et exécutez-le.

Exercice: niveau 3

1. Écrivez un exemple pour différents types de données Python tels que le nombre (entier, float, complexe), String, Boolean, List, Tuple, Set et Dictionary.
2. Trouvez une distance euclidienne entre (2, 3) et (10, 8)

Félicitations!