I. Créer une variable

Python est un langage de programmation fantastique pour les débutants et les experts.
C'est la langue de choix pour de nombreuses entreprises et un choix populaire pour les projets personnels.

Vous pouvez l'utiliser pour effectuer des tâches, progresser dans l'analyse des données, l'apprentissage automatique et bien plus encore.

Peu importe la complexité d'un programme, il commence par une seule ligne de code. Cette première ligne est habituellement une **variable**.

Les programmes utilisent la variable pour **se souvenir de l'information**. Comme le déplacement des boîtes, les variables ont du contenu et des noms qui nous disent ce qu'il y a à l'intérieur.

Pour créer une variable, nous devons lui donner un nom. Les noms de variable doivent être des mots simples et, par conséquent, n'ont pas d'espaces.

Tapez la variable *city* pour commencer.

city

Si nous voulons un nom de variable avec plusieurs mots, nous utilisons un *snake case*.
Un *snake case* signifie utiliser _ pour connecter les mots supplémentaires.

Par exemple : *home_city*

Pour nous aider à comprendre ce qu'est une variable, nous choisissons des noms descriptifs.

Par exemple : *home_city_province*

Exercices

1. À quoi servent les variables utilisées par les ordinateurs?
a) À dessiner des images à l'écran
b) À se souvenir de l'information

2. Qu'est-ce qui cloche avec ce nom de variable?
high score
a) Il y a un espace entre les mots
b) Il contient une lettre minuscule

3. À quoi sert le *snake case* ?
a) Pour créer des noms de variables avec des espaces
b) Créer des noms de variables avec plusieurs mots.

4. Pourquoi donnons-nous des noms descriptifs de variables comme *city* ou *population* plutôt que *c* ou *p* ?
a) Pour les rendre plus rapides à taper.
b) Pour nous aider à comprendre ce qu'ils contiennent.

5. Choisissez le nom de variable le plus descriptif : *high_score*, *h_s* ou *hs* ?
Réponse : *high_score*

6. Ajoutez les bons signes pour créer la variable de *snake case* : _, +, _
Réponse : *total_cash_amount*

7. Créez un nom de variable descriptif.

Réponse : *number_of_requests*

8. Finissez de nommer la variable *friend_* avec *requests*.
Réponse : *friend_requests*

II. Utiliser des variables

Les variables sont appelées variables parce que les valeurs qu'elles stockent peuvent changer. Nous pouvons mettre à jour *statut* en utilisant = et en lui donnant une nouvelle valeur.

```
statut = "Regarder Netflix"
statut = "Se détendre à la plage"
print(statut)
```

Nous pouvons mettre à jour les variables aussi souvent que nous le voulons. Essayez-le vous-même : changez la valeur de *statut* à *"Nouvelles données requises"* .

script.py
```
statut = "Incomplet"
statut = "Complet"
print(statut)
statut = "Nouvelles données requises"
print(statut)
```

Résultat
```
Complet
Nouvelles données requises
```

Nous pouvons aussi donner aux variables les valeurs d'autres variables. Ici, nous pouvons donner à la variable *new_status* la valeur de *default_option* .

script.py

```
default_option = "Charger"
new_status = "Télécharger"
new_status = default_option
print(new_status)
```

Résultat
```
Charger
```

Lorsque nous mettons à jour une variable, elle oublie sa valeur précédente. Ici, nous pouvons afficher la variable *statut* deux fois et voir comment sa valeur se met à jour.

script.py
```
statut = "Jouer au football"
Print(statut)
statut = "Promener le chien"
Print(statut)
```

Résultat
```
Jouer au football
Promener le chien
```

Exercices

1. Laquelle de ces lignes de code met à jour la variable *statut* ?
a) statut == "Travail"
b) statut = "Travail"

2. Qu'est-ce que ce code affiche dans la console ?
statut = "Regarder Netflix"
a) statut
b) Regarder Netflix

3. Qu'est-ce qui cloche avec le code?
nom = "Lori"
nom = "Joe"
a) Rien
b) La même valeur est stockée dans la variable

4. Qu'est-ce que le code affiche dans la console ?
analyse = "Moyenne"
analyse = "Médiane"
print(analyse)
a) Moyenne
b) Médiane

5. Changer la valeur de la variable *température* à *"100 degrés"* .
température = "0 degrés"
température = "100 degrés"
print(température)
Résultat
100 degrés

6. Mettez à jour la variable *statut* à *"Écrire du code"* .
statut = "Remplir une feuille de calcul"
statut = "Écrire du code"
Print(statut)
Résultat
Écrire du code

7. Mettez à jour la variable *devise* à *"Dollar"* .
devise = "Euro"
devise = "Dollar"
Print(devise)
Résultat
Dollar

8. Changer la valeur de la variable *statut* à *"Terminé"*.
statut = "Inachevé"
statut = "Terminé"
Print(statut)
Résultat
Terminé

III. True ou False

Il y a une valeur spéciale qui n'est **ni** une chaîne **ni** un nombre : *True*.

Il n'y a pas de guillemets, et ce n'est pas une valeur numérique.

True est idéal pour les solutions comme la vérification d'une fonctionnalité activée ou si les données sont disponibles. Nous pouvons le voir ici lorsque nous définissons *powered_on* à *True*.

script.py
```
powered_on = True
```

Nous pouvons mettre *True* dans une variable comme une chaîne ou un nombre.
L'affichage fonctionne également, comme lorsque nous affichons *correcte* ici.

script.py
```
correcte = True
print(correcte)
```

Résultat
```
True
```

False est une autre valeur spéciale et le contraire de *True*.

Nous pouvons enregistrer *False* dans la variable *statut* et afficher *statut* dans la console.

script.py
```
print("Charger les données")
statut = False
print(statut)
```

Résultat
```
Charger les données
False
```

Exercices
1. Quelle est la bonne utilisation des valeurs *True* et *False* ?
a) Indiquer si une fonction est activée ou désactivée
b) Stockage de valeurs de 1 à 5

2. Pourquoi *False* n'est-il pas une chaîne ?
```
Télécharger = False
```
a) Il est stocké dans une variable
b) Il n'y a pas de guillemets.

3. Choisissez celui qui est le mieux pour montrer un utilisateur non abonné à partir d'un service.
```
a) souscrit = False
```
b) souscrit = True

4. Pourquoi *"False"* n'est-il pas la même chose que *False* ?
a) Il y a des guillemets autour, donc "False" est une chaîne
b) "False" et False sont les mêmes

5. Afficher l'état du paramètre de *mise à jour automatique* en affichant *False* dans la console.
```
print("Mise à jour automatique")
print(False)
```
Résultat
```
Mise à jour automatique
False
```

6. Créez la variable *auto_save* et stocker *True* à l'intérieur.
script.py
```
auto_save = True
```

7. Créez la variable pour empêcher la déconnexion et stocker *False* à l'intérieur.
script.py
prevent_logout = False

8. Stocker une variable qui sera utilisée dans les calculs en codant *True*.
script.py
calculate_average = True

IV. Contrôle de l'égalité des nombres

Nous avons appris comment créer et stocker des valeurs, mais comment les comparer ?
Par exemple, si le code PIN d'un utilisateur correspond au code PIN enregistré.

script.py
saisi_pin = 5448
sauvegarde_pin = 5440

Pour comparer si deux nombres sont identiques, nous utilisons l'**opérateur égalité**, ==.

script.py
5 == 5

Lors de la comparaison, il n'y a que deux résultats : *True* ou *False*.

Lorsque nous comparons les mêmes nombres avec l'**opérateur d'égalité**, le résultat est *True*.

script.py
print(10 == 10)

Résultat
True

Lorsque nous comparons les différents nombres avec l'**opérateur d'égalité**, le résultat est *False*.
Comme ici avec la comparaison de *9* à *10*.

script.py
print(9 == 10)

Résultat
False

Exercices

1. Que devons-nous utiliser pour vérifier si deux nombres sont égaux?
a) ==
b) =

2. Que montre ce code dans la console ?
print(10==11)
a) False
b) True

3. Quand pourrions-nous vérifier si deux nombres sont égaux?
a) Lors de la vérification si une variable est exactement égale à 10
b) Lorsque la vérification d'une variable est supérieure à 50

4. Qu’est-ce que ce code affiche dans la console.
print(5 + 13)
print(5 == 13)
a) 18 puis 513
b) 18, puis False

5. Vérifiez si la valeur des votes est égale à 11: =, *11*, =
votes = 10
print(votes == 11)
Résultat
False

6. Ajouter l’opérateur d’égalité : =, ==
print(99 == 100)
Résultat
False

7. Affichage faux dans la console : *10*, *13*
print(10 == 13)
Résultat
False

8. Vérifier si ces nombres sont identiques par codage == .
print(100 == 100)
Résultat
True