Introduction à Python

Table des matières

Contexte

Durée : 2h

Environnement de travail : Repl.it

Pré-requis : Aucun

[cf. yd6X2ErU]

Les études sur l'utilisation des langages de programmation ne manquent pas, et Python figure systématiquement dans les plus utilisés au monde.

On retrouve Python dans le scripting Linux, l'intelligence artificielle (via des librairies/framework comme PyTorch, Keras, TensorFlow, etc.), le calcul scientifique (avec numpy, pandas, scikit, etc.) et le Web (via Django, Pymamid, etc.).

Sa facilité de prise en main par rapport à d'autres lui ont permis de se faire une place dans tous ces domaines et sa communauté gigantesque lui offre un écosystème difficilement atteignable.

Histoire, spécificité, licence

[cf. LhyD8j83]

Objectif

- Découvrir ce qui caractérise Python.

Mise en situation

Python est un langage de programmation interprété et multi-paradigmes. En particulier, il gère les paradigmes impératif, fonctionnel et objet.

Il a été conçu avec l'idée d'offrir des outils de haut niveau et une syntaxe simple à apprendre et utiliser. Ses possibilités ont beaucoup évolué depuis sa première version en 1991.

Histoire

- En 1989, Guido van Rossum lance le développement d'un nouveau langage, le Python, pour l'aider dans ses recherches. Une première version publique de Python sort en 1991.
- En 1995, Van Rossum continue son travail au États-Unis et travail sur un projet visant à faire du Python un langage d'apprentissage.
- Au cours des années suivantes, l'équipe de développement de Python va changer plusieurs fois d'organisation et la version 2.0 aura été atteinte.
- Enfin, c'est en 2001 que la Python Software Foundation est créée pour la sortie de la version 2.1.

Python Software Foundation et licence

C'est une organisation sans but lucratif et entièrement dédiée au Python. Elle est responsable du développement, de la protection intellectuelle et de conférences (PyCon) autour du langage.

Elle a aussi créée la licence *« Python Software Foundation License »* qui est une licence libre semblable à la licence BSD et compatible avec la GPL (GNU General Public License) (sauf pour les versions 1.6 à 2.1).

Quelques spécificités

Remarque

- **Typage dynamique** : l'interpréteur décide du type de chaque variable au moment où il la rencontre.
- **Multi-plateformes** : il existe des interpréteurs Python pour générer du code machine sur tous les systèmes d'exploitation connus.
- **Multi-paradigmes** : ce langage favorise la programmation impérative mais offre des éléments importants de la programmation fonctionnelle (fonctions `map`, `reduce`, etc.) et la possibilité d'instancier des objets à partir de classes.

Syntaxe ***Exemple***

```
1 def print_sum(x, y):
2     print("%d + %d = %d" % (x, y, x+y))
3
4 a = 42
5 b = 20
6
7 print_sum(a, b)
```

Ce code :

- Définit une fonction `print_sum` qui affiche la somme de deux termes ;
- Appelle cette fonction avec les paramètres `a` et `b`.

Notez que les variables `a` et `b` ne sont pas préalablement déclarées ni typées.

Notez également le rôle de l'indentation pour délimiter le contenu des blocs.

À retenir

- Python se veut simple mais puissant.
- Son développement a commencé en 1989 et est toujours fortement actif avec une grande communauté de développeurs.
- Python est multi-paradigme et offre des outils haut niveau.

Exercice : Appliquer la notion : enquête Wikipédia

Exercice

Depuis cette page Wikipédia.[1]

Quel est la licence actuelle du Python ?

- ○ AGPL
- ○ Mozilla Public License
- ○ Python Software Foundation License
- ○ Son code source est propriétaire

Exercice

Quel type/objet n'existe pas ?

- ☐ int
- ☐ dict
- ☐ None
- ☐ double

[1] https://fr.wikipedia.org/wiki/Python_(langage)

Repl et éditeurs

[cf. Ts7hWwCz]

Objectif

- Découvrir comment utiliser un éditeur pour Python.

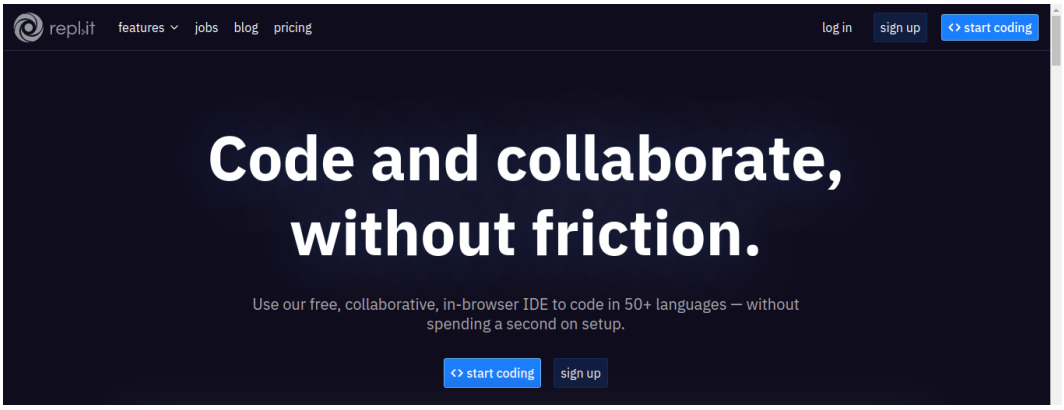
Mise en situation

L'éditeur de texte est un des outils importants dans le quotidien du développeur. Les codes sources des programmes sont de simples fichier textes, qui ont besoin d'être écrit et manipulés dans un logiciel adapté. Quel que soit le langage, un éditeur peut apporter bien plus que la possibilité d'écrire du code et va accompagner le développeur dans son travail, par exemple en affichant de la documentation dynamiquement, ou en appliquant des corrections de syntaxe. Nous allons découvrir ici un éditeur en ligne, Repl.it.

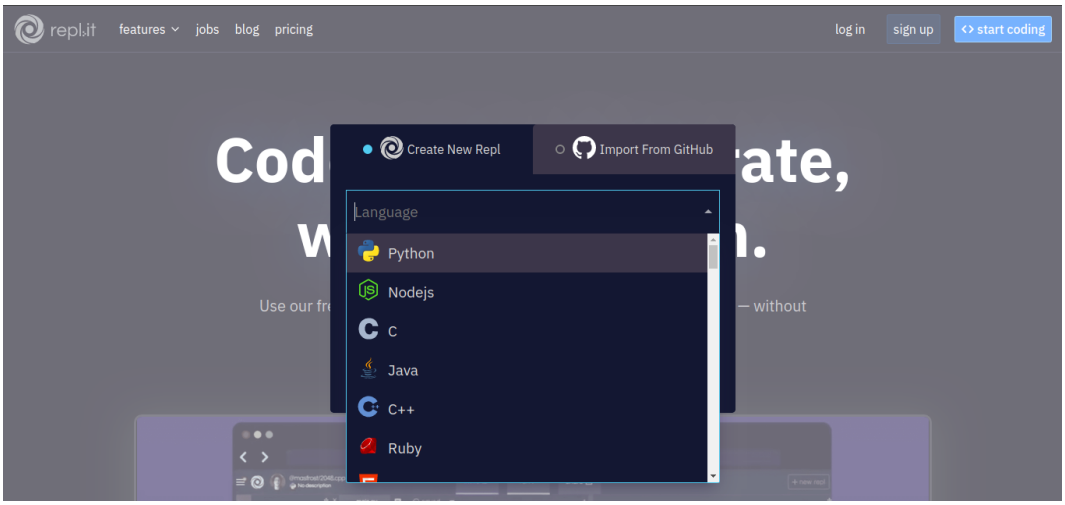
Les éditeurs et IDE

Pour commencer rapidement à faire du Python sans rien installer, il existe des éditeurs/IDE en ligne tel que Repl.it.

Sur la page d'accueil, le bouton « `<> start coding` » permet de créer un projet.



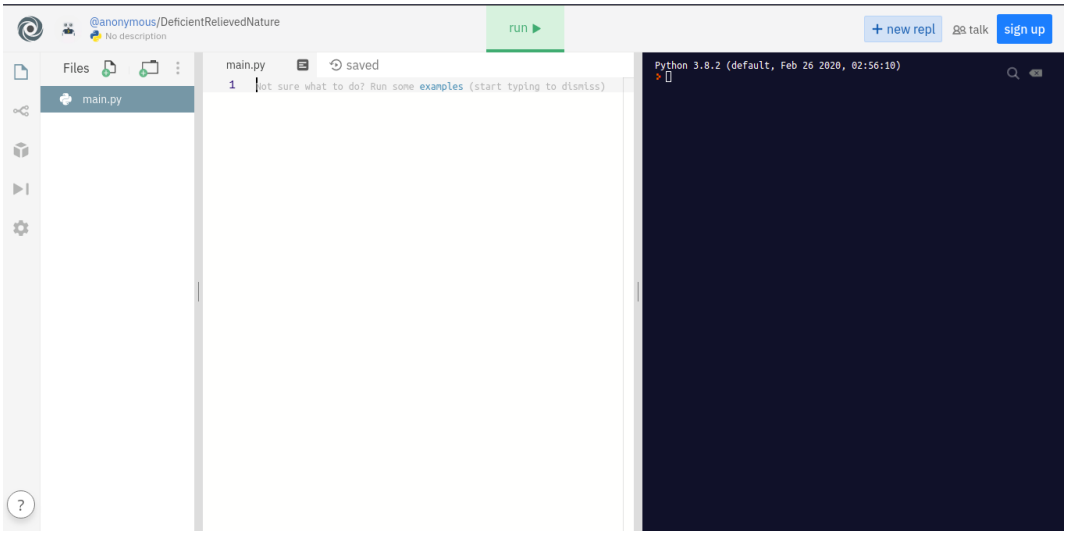
Il est ensuite possible de sélectionner le Python et confirmer avec `Create Repl`.



Une fois sur la fenêtre Repl on distingue trois grandes colonnes. En partant de la gauche :

1. Les paramètres de l'éditeur et les fichiers du projet (au départ il n'y a que `main.py` pour un projet Python).
2. L'éditeur pour écrire son code.
3. Une console avec un interpréteur Python. Cette console servira d'entrée et sortie standard du programme.

Sur la barre horizontal du haut, le bouton le plus important est le bouton `Run` qui permet de lancer l'interprétation/l'exécution du programme.



À retenir

- Les éditeurs/IDE sont parmi les outils les plus importants d'un développeur.
- Repl.it est un IDE en ligne.

[cf. emW9s0mu]

Exercice : Appliquer la notion

Question

Sur Repl.it[1], créer un nouveau projet Python et exécuter le code suivant.

```
1 math_grade = 14
2 french_grade = 17
3
4 average = (math_grade + french_grade) / 2
5 print("Votre moyenne est de %.2f" % average)
```

Que voit-on sur le terminal ?

Indice :

Le bouton `Run` permet d'exécuter le code.

[1] https://repl.it/

Syntaxe

[cf. C8rm3mvK]

Objectif

- Découvrir la syntaxe de base du Python.

Mise en situation

Python a été conçu en gardant à l'esprit la lisibilité et sa syntaxe reflète cette contrainte. C'est un langage qui ne s'alourdit pas avec des accolades ou des caractères de fin de ligne, et qui n'utilise des parenthèses que lorsque cela est nécessaire. Pour gérer les différents blocs de code, tout se base sur l'indentation. De ce fait un code en Python valide est très facilement lisible par un humain. C'est sans doute pour cette raison qu'il est particulièrement apprécié dans les milieux qui ne viennent pas directement de l'informatique, comme le milieu de la recherche scientifique.

Typage dynamique

Fondamental

Une variable permet de stocker une donnée dans la mémoire du programme.

Le typage du Python étant **dynamique**, il n'y a pas de contrainte sur ce qu'une variable peut contenir et le type de son contenu peut être modifié à la volée.

Les chaînes sont délimitées par des apostrophes simples `'` ou des doubles `"`.

Variables

§ *Syntaxe*

Une valeur est affectée à une variable via l'opérateur d'affectation =.

```
number_of_bytes = 18005  # Entier

number_of_bytes = number_of_bytes / 2  # Décimal

# Ce code est valide.
```

Constantes

Complément

Une constante est une variable qui, une fois déclarée avec une valeur, ne peut plus changer son contenu.

Ce type de variable n'existe pas en Python mais la convention veut qu'un développeur Python écrive ses constantes en majuscules.

```
LIGHT_SPEED = 299792458

# Cette variable est modifiable mais elle doit être considérée comme une constante
par les développeurs.
```

Entrée/sortie

§ *Syntaxe*

- La fonction `print()` permet d'afficher une variable dans la console.
- La fonction `input()` permet de saisir une variable via la console.

? *Exemple*

```
1 message = input("Écrivez-moi un message : ")
2 print ("J'ai bien reçu votre message, il disait : " + message)
3
```

Expressions

§ *Syntaxe*

Une expression est une opération qui retourne une valeur. Cette opération peut être une combinaison d'opérateurs, variables et fonctions qui, lorsque elle est évaluée produit un résultat.

Le Python se base sur une syntaxe d'expressions inspiré des notation mathématique, comme dans la plupart des autres langages de programmation comme le C ou le JavaScript.

```
1 result = variable/value operation value/value
```

? *Exemple*

```
1 image_weight = 49583  # Taille en octet d'un image
2 mask_weight = 3000  # Taille en octet d'un masque à appliquer sur une image
3
4 # Taille finale que l'on peut utiliser dans la suite
5 # du programme
6 final_weight = image_weight + mask_weight
```

`image_weight + mask_weight` est l'expression (composée d'un opérateur d'addition) dont le résultat est stocké dans la variable `final_weight`.

Indentation

Attention

Contrairement à beaucoup de langages tels que le C ou le JavaScript, la délimitation des blocs d'instructions se fait par l'**indentation** et non via des accolades ou des mots clés.

Cette pratique force l'écriture d'un code lisible, contrairement à d'autres langages où l'indentation est optionnelle.

En contre-partie, une erreur de présentation conduira à une erreur d'interprétation.

Méthode

- On utilise des espaces pour indenter le code (et non des tabulations).
- Le standard de codage PEP8 préconise d'utiliser quatre espaces pour chaque tabulation.

Indentation

§ *Syntaxe*

```
1 price = 22
2
3 if price < 15:
4     print("On envoie une alerte de prix à l'utilisateur")
5 else:
6     print("C'est trop cher, on attend un prix plus bas")
```

À retenir

- Python est d'abord un langage impératif.
- Il met en avant la lisibilité du code.
- Les indentations ont une signification.
- Sa syntaxe est proche à celle d'autres langages comme JavaScript ou C.

[cf. fXYFD6yB]

Exercice : Appliquer la notion

Sur Repl.it[1] :

Question 1

Écrire les instructions permettant de :

- Déclare une variable `name` dont le contenu est la chaîne *Alpha*.
- Affiche *Bienvenue sur notre logiciel, [name]*, `[name]` étant le contenu de la variable `name`.

Indice :

La fonction `print` permet d'afficher un message dans la console.

L'opérateur de concaténation est le `+`.

Question 2

Demander à l'utilisateur d'écrire son nom.

Indice :

La fonction `input` permet de demander une donnée à l'utilisation.

Question 3

Combiner les deux codes précédents pour souhaiter la bienvenue à l'utilisateur.

[1] https://repl.it/

Structures de base

[cf. zUZkWLSb]

Objectif

- Se familiariser avec les structures de base du Python.

Mise en situation

Python propose des structures de bases classiques, celles que l'on retrouve dans de nombreux langages. Par exemple les structures conditionnelles à base de `if` et `else`, ou encore les boucles à compteur ou à condition. Il intègre aussi les fonctions prédéfinies classiques, comme celles pour interagir avec les entrées et sorties, et des fonctions de traitement des structures comme les tableaux.

L'affectation

Syntaxe

L'affectation permet d'assigner une valeur à une variable.

```
variable = valeur
```

Exemple

```
number_of_bytes = 4096
image_to_process = "milky_way.png"
```

if

Syntaxe

L'instruction `if` permet d'exécuter un bloc d'instructions si une condition est vérifiée.

```
if condition:
    instructions
```

Exemple

```
grade = 16

if grade >= 16:
    print("Très bon travail")
```

Complément

Les conditions en Python sont :

- `a == b` : a est égal à b
- `a != b` : a est différent de b
- `a < b` : a est strictement inférieur à b
- `a > b` : a est strictement supérieur à b

- `a <= b` : a est inférieur ou égal à b
- `a >= b` : a est supérieur ou égal à b

Toutes ces conditions sont combinables les opérateurs logiques « *ET* » et « *OU* », qui s'écrivent `and` et `or`.

elif et else

Complément

Lorsque la condition du `if` n'est pas vérifiée, le programme regarde si il y a d'autres branchements conditionnels `elif` (*sinon si*).

De la même manière, si aucune des conditions du `if` et des `elif` n'est vérifiée, le programme exécutera le branchement `else` s'il existe.

```
grade = 16

if grade >= 16:
    print("Très bon travail")
elif grade < 16 and grade >= 13:
    print("Bon travail")
elif grade < 13 and >= 10:
    print("Passable")
else:
    print("À revoir")
```

Rappel

Une boucle sert à répéter plusieurs fois les mêmes instructions.

while

Syntaxe

L'instruction `while` permet d'entrer (et de rester) dans une boucle tant qu'une condition est vérifiée.

```
while condition:
    instructions
```

Exemple

```
stay_in_loop = True

while stay_in_loop:
    print("Voulez-vous sortir de la bouche ? (o/n)")
    user_input = input()

    if user_input == "o" or user_input == "O":
        stay_in_loop = False
```

Cette boucle va demander, à chaque itération, si l'utilisateur veut sortir de la boucle. Si oui, alors la variable `stay_in_loop` passe à la valeur `False` et le programme sort de la boucle.

for

Syntaxe

L'instruction `for` permet d'exécuter un bloc d'instructions un nombre précis de fois. Elle a besoin d'une variable d'itération et d'une séquence de valeurs à parcourir.

```
for variable in range(valeur1, valeur2):
    instructions
```

À chaque itération, la variable d'itération devient l'élément courant dans la séquence parcourue ; `range()` est une fonction qui renvoie une liste de `valeur1` incluse à `valeur2` exclue.

Exemple

```
1 for i in range(1, 10):
2     print('%s x 9 = %s' % (i, i * 9))
3
```

À retenir

- Python propose une structure `if elif else` pour avoir plusieurs branches d'exécution.
- Python dispose de la boucle `while` pour les nombres d'itération indéfinis et `for` pour parcourir une séquence.

[cf. YLHW6duM]

Exercice : Appliquer la notion

Question

À l'aide d'une boucle et d'une structure alternative, écrire un programme qui à partir des variables entières `start` et `stop` renvoie une liste contenant les nombres pairs entre ces deux valeurs.

Indice :

L'opérateur « % » renvoie le reste d'une division (exemple : 8 % 2 = 0).

Un nombre est pair s'il vaut 0 modulo 2.

Indice :

```
start = 5
stop = 28
...
```

Indice :

```
start = 5
stop = 28
for i in range(...):
    if ...:
        print (i)

```

Bonnes pratiques

[cf. 9iKtsn57]

Objectifs

- Connaître les conventions de styles du Python ;
- Connaître la PEP 8.

Mise en situation

Comme tous les langages, Python offre beaucoup de libertés aux développeurs. Cependant, ces derniers doivent être en mesure de travailler ensemble sur des projets communs. Les libertés offertes par le Python viennent donc ajouter quelques responsabilités aux développeurs.

La PEP 8

Les PEP - Python Enhancement Proposals (python.org/dev/peps[1]) - sont des propositions pour améliorer le langage. Elles permettent d'avoir un système de suivi des propositions pour savoir ce qui a été fait et ce qu'il y a à faire dans le développement du langage lui-même.

- PEP 8 (python.org)[2]

Fondamental

La PEP 8 est une des plus anciennes propositions (2001) et elle introduit le style officiel à respecter lorsque l'on code en Python. En voici quelques éléments de base :

- Indentation : un niveau d'indentation correspond à 4 espaces.
- Longueur d'une ligne : ne pas dépasser 79 caractères.
- Conventions de nommage : les noms de fonctions et de variables doivent être en *snake_case*, c'est à dire en minuscule, avec les mots séparés par *underscore*, comme `une_variable`. Les constantes sont en *SNAKE_CASE* majuscules.
- Espaces : on place des espaces autour des opérateurs et des arguments de fonction.

Exemple

```python
# INCORRECT
def sum(a,b):
  return a+b
someVariable=42
print(sum(someVariable,20))

# CORRECT
def sumBis(a, b):
    return a + b
some_variable = 42
```

[1] https://www.python.org/dev/peps/
[2] https://www.python.org/dev/peps/pep-0008/

```
sumBis(some_variable, 20))

```

Les commentaires

Les *block comments* :

- commencent par « # »,
- sont suivis d'un espace et doivent être composés de phrases entières,
- peuvent être chaînés sur plusieurs lignes pour une explication plus longue.

Les *inline comments* :

- commencent aussi par « # » mais se trouvent directement sur la ligne de l'instruction commentée,
- sont séparés par au moins 2 espaces de la fin de l'instruction,
- doivent être utilisés pour donner une information non triviale (un effet de bord par exemple).

Les *docstrings* :

- permettent de documenter le code sur plusieurs lignes en plaçant les commentaires entre « """ »,
- sont placés directement sous les signatures de fonctions, méthodes et classes pour expliquer à quoi ces derniers servent.

Exemple

```
def isCSV(filename):
    """
    Ceci est un docstring pour expliquer la fonction autour.

    Cette fonction prend en entrée un nom de fichier.
    Et retourne un booléen indiquant si le fichier est un csv.
    """
    if filename[-3:] == "csv":  # Inline comment
        return True
    return False

# Block comment
print(isCSV("employees.csv"))
```

À retenir

- Python offre de nombreuses libertés.
- L'utilisation de la convention PEP 8 permet de rendre le code plus facile à lire et comprendre pour tout le monde.

[cf. F9O1ngcH]

Exercice : Appliquer la notion

Exécuter ce code pour vous assurer qu'il fonctionne :

```
def print_name():
"""
Greets the user.

Greets the user by asking their name.

"""
print("Hello, what's your name?")
Name = input()
print("Welcome"+Name)

print_name()
```

Question

Corriger le style pour adopter la convention PEP 8.

Indice :

Regarder les conventions de nommage des variables.

Faites attention aux indentations fausses.

Attention aux espaces entre les opérateurs.

Essentiel

[cf. HQZFjiAR]

Nous avons vu que Python est un langage très populaire du fait de sa simplicité de prise en main. Il offre les mêmes structures et fonctions de base que les langages de programmation les plus répandus, comme JavaScript, mais sa syntaxe est plus accessible. Sa communauté est importante, ce qui le rend très utile pour débuter, car les ressources ne manquent pas pour aider à progresser.

Cependant, cette simplicité n'en fait pas un simple langage d'apprentissage. Il est utilisé dans de nombreux projets sérieux, comme par exemple dans la recherche scientifique ou au niveau des algorithmes d'intelligence artificielle.

Quiz

Exercice 1 : Quiz - Culture

Exercice

De quand date la première version publique de Python ?

- ○ 1989
- ○ 1991
- ○ 2001
- ○ 2005

Exercice

De quand date la PEP 8 ?

- ○ 1989
- ○ 1991
- ○ 2001
- ○ 2005

Exercice

Quelles sont des caractéristiques du Python ?

- ☐ Récupérateur de mémoire (*garbage collector*)
- ☐ Typage statique
- ☐ Multi-paradigmes
- ☐ Licence libre

Exercice 5 : Quiz - Méthode

Exercice

Comment déclarer une variable `ma_variable` contenant un entier ?

- ☐ `int ma_variable;`
- ☐ `ma_variable = 42`
- ☐ `int ma_variable = 42`
- ☐ Aucune de ces réponses

Exercice

Lesquels de ces éléments est à privilégier pour stocker les chaînes "Python", "Vipère", "Anaconda" afin d'itérer dessus ultérieurement ?

- ○ Un dictionnaire
- ○ Une liste
- ○ Une fonction
- ○ Une exception

Exercice

Quel type de structure est adéquate pour implémenter le menu suivant :

```
***** Menu *****
0. Sortir
1. Afficher la liste des chansons programmées
2. Programmer une nouvelle chanson
3. Déprogrammer une chanson
```

- ○ Une instruction `while`
- ○ Une instruction `for`
- ○ Une instruction `if`
- ○ Aucune de ces réponses

Exercice 9

Appairez les termes déclarations suivantes :

`VitesseMax = 300` `VITESSE_MAX = 300` `vitesse_max = 300`

`vitesse-max = 300`

Variable conforme à la convention PEP 8	Constante conforme à la convention PEP 8	Déclaration non conforme à la convention PEP 8

Exercice 10 : Quiz - Code

Exercice

```
note = 18

if note >= 18:
    print("A")
if note <= 18 and note >= 15:
    print("B")
elif note < 15 and note >= 13:
    print("C")
elif note < 13 and note >= 10:
    print("D")
elif note < 10 and note >= 8:
    print("E")
else:
    print("F")
```

Quelles lettres sont affichées sur la sortie standard ?

- [ ] A
- [ ] B
- [ ] F
- [ ] Le programme renvoie une erreur.

Exercice

```
note = 18

if note >= 18:
    print("A")
elif note <= 18 and note >= 15:
    print("B")
elif note < 15 and note >= 13:
    print("C")
elif note < 13 and note >= 10:
    print("D")
elif note < 10 and note >= 8:
    print("E")
else:
    print("F")

```

Quelles lettres sont affichées sur la sortie standard ?

- [ ] A
- [ ] B
- [ ] F
- [ ] Le programme renvoie une erreur.

Exercice

```
1 note = 18
2
3 if note > 18:
4     print("A")
5 elif note <= 18 and note >= 15:
6     print("B")
7 elif note < 15 and note >= 13:
8     print("C")
9 elif note < 13 and note >= 10:
10    print("D")
11 elif note < 10 and note >= 8:
12    print("E")
13 else:
14    print("F")
15
```

Quelles lettres sont affichées sur la sortie standard ?

- [ ] A
- [ ] B
- [ ] F
- [ ] Le programme renvoie une erreur.

Crédits des ressources

p. 3

http://creativecommons.org/licenses/by/4.0/fr/

p. 7

http://creativecommons.org/licenses/publicdomain/4.0/fr/

p. 7

http://creativecommons.org/licenses/publicdomain/4.0/fr/

p. 8

http://creativecommons.org/licenses/publicdomain/4.0/fr/