NSI programmation

Spécifier programmes

qkzk

Python

Présentation de Python

Python est un langage de programmation crée au début des années 90 par Guido Van Rossum.

Python est choisi pour

- le lycée : informatique, maths, physique, snt
- les classes prépa et les licences informatiques

C'est un langage qui facile d'accès, idéal pour les débutants.

Cependant Python est un langage puissant et complet qui a de nombreux usages dans l'industrie informatique.

Caractéristiques de Python

- Python est *gratuit* et ses sources sont *ouvertes*. Tout le monde peut les consulter et les améliorer s'il le souhaite.
- Python est un *langage haut niveau*. Cela signifie qu'on est éloigné de ce qui se passe réellement dans la machine et qu'on peut se concentrer sur le déroulement du programme en lui même.

- Python est un langage *interprété* et non compilé. Cela signifie qu'à chaque exécution d'un programme Python, un programme appelé interpréteur traduit le code en *bytecode* exécutable par la machine. Le code est ainsi facile à éditer et à corriger.
- Python est à la fois *multi-plateforme* (il fonctionne sur tous les systèmes courants) et *multi-paradigme* (il existe plusieurs manières d'écrire un programme qui accomplisse le même résultat)
- Python est doté d'un *typage dynamique*. Contrairement au C ou à Java, il n'est pas nécessaire de déclarer le type d'une variable. Celui-ci peut changer durant l'exécution du programme.
- La syntaxe de Python diffère de celles inspirées du C. C'est *l'indentation qui décrit la structure d'un programme.*
- Python dispose de plusieurs interpréteurs mais le plus courant est *CPython* écrit en C.
- Python dispose de deux versions majeures partiellement incompatibles : Python 2.7 et *Python 3*. Nous utiliserons cette année la dernière version stable de Python 3

Quelques éléments de syntaxe

La syntaxe de Python est conçue pour être facilement lisible.

Les commentaires sont précédés d'un symbole `#`

```
a = 3 # un commentaire
```

Tests et conditions

La structure (les blocs) sont indiqués par une indentation (généralement 4 espaces)

Par exemple :

```
a = 3
if a > 2:
  print(f"{a} est plus grand que 2")
elif a == 2:
  # affecter =
  # comparer ==
  print(f"{a} vaut 2")
else:
  print(f"{a} est inférieur à 2")
```

Les espaces après `if` et `else` indiquent ce qui doit être exécuté si ces conditions sont vérifiées.

Boucles

Il existe deux types de boucles `for` et `while`

La syntaxe de `while` est similaire à celle du C :

```
a = 1
while a < 5:
  # faire quelquechose
  a = a + 1
```

La syntaxe de `for` est différente. On parcourt un objet itérable

```
liste = [1, 2, 3, 4, 5, 6]
for nombre in liste:
  if nombre % 2 == 0:
    # nombre % 2 est le reste de la division euclidienne par 2
    print(f"{nombre} est pair")
  else:
    print(f"{nombre} est impair")
```

Les fonctions

Une fonction est un morceau de code qui peut être appelé et exécuté plusieurs fois. Une fonction prend des *paramètres* en entrée et *retourne* un résultat.

Toutes les fonctions Python retournent une valeur avec `return`.

Si rien n'est indiqué après `return` ou si ce mot-clé est omis, la fonction retourne `None`

```
def carre(n):
  '''
  calcule le carré d'un nombre
  @param n: (number)
  @return: (number)
  '''
  return n ** 2
```

On *spécifie* une fonction en indiquant dans une chaîne de caractères sur plusieurs lignes :

- ce qu'elle fait,
- ses paramètres et leur type
- ce qu'elle renvoit

Objets simples

Les *types* courants en Python sont :

- `int` : les entiers (0, -1, 234567890 etc.)
- `float` : les "nombres à virgules" (0.2, 1234.1234 etc.)
- `str` : les chaînes ce caractères (`"a"`, `"bonjour David59"`)
- `bool` : `True`, `False`
- `Nonetype` : `None` (rien)

Objets complexes

Ils sont généralement *itérables*, on peut les parcourir élément par élément.

- `list` : `[1, 2, 3]` une liste (ou un tableau) d'objets. Mutable.
- `tuple` : `(1, 2, 3)` une liste non mutable (immuable) d'objets.
- `dict` : `{"nom" : "David", "tel" : "0612345678"}`. tableau associatifs ou dictionnaire. Mutable

Il en existe de nombreux autres que nous rencontrerons plus tard.

Librairies

On importe une librairie avec `import`

```
from math import pi
def circonference(rayon):
  '''
  Calcule la circonference d'un cercle
  @param rayon: (number)
  @return: (float)
  '''
  return 2 * pi * r
```

Python est fourni avec une *librairie standard* très riche qu'il est rarement nécessaire d'étendre.

Cela est néanmoins facile avec le programme `pip` :

```
$ pip install numpy
```

Nous utiliserons notamment :

- `numpy` : calculs numériques,
- `matplotlib` : constructions de figures mathématiques,
- `pygame` : jeux vidéos,
- `flask` : création d'un site web

Adoption

Python est massivement utilisé par les professionnels. Depuis 2014 c'est le langage qui connait la plus forte croissance dans la communauté des développeurs.

C'est le second langage le plus cité après `JavaScript` comme étant celui qu'ils préfèrent utiliser.

Parmi les applications fréquentes de Python citons :

- L'intelligence artificielle avec les librairies `panda` `sklearn` et `tensorflow`,
- Le calcul numérique avec `numpy` et `matplotlib`,
- Les logiciels (fenêtres etc.) avec `QT` et `Tkinter`,
- Les tests. Les développeurs doivent s'assurer du bon fonctionnement et de l'efficacité de leurs programmes. De nombreux tests sont écrits en Python.

Parmi les entreprises et projets qui utilisent Python citons :

- Google,
- La NASA,
- Industrial Light & Magic (effets spéciaux de Disney, LucasFilm etc.)
- Netflix,
- LibreOffice etc.

Citons Katie Bouman, célèbre pour avoir présenté la première image d'un trou noir et qui travaille au MIT.

Figure 1: blackhole

On peut voir qu'elle est en train de programmer un script Python...