Python pour l'application des math.

Sv

11 octobre 2022

Table des matières

1 Contrôle du flux d'exécution

2 Instructions répétitives

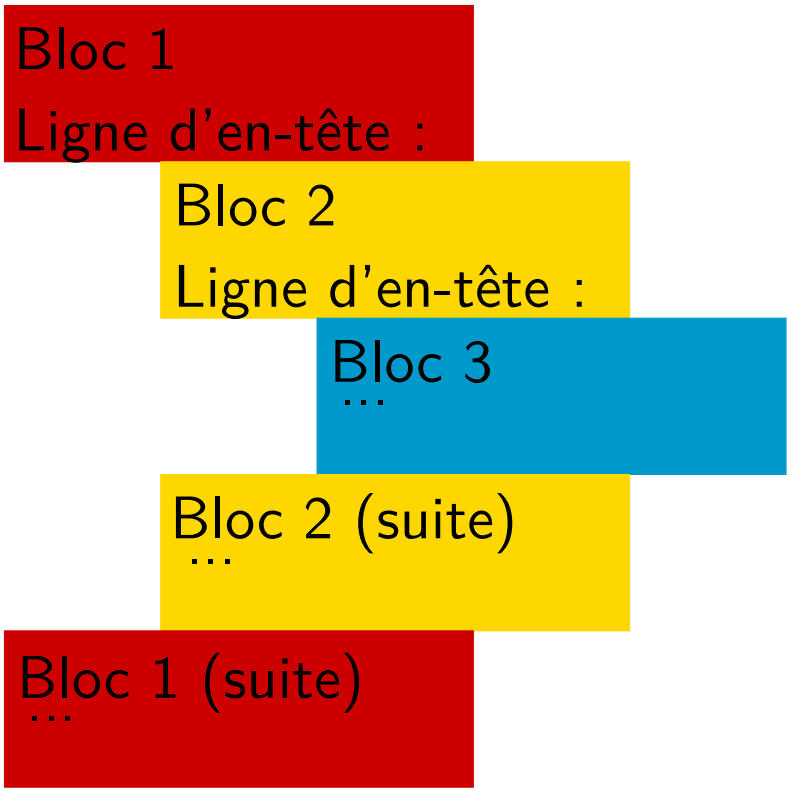
3 Les fonctions

Un bloc d'instructions

Instructions composées

```
Ligne d'en-tête:
    première instruction
    ....
    ....
    dernière instruction
```

Plusieurs blocs d'instructions



Instruction if

```
a = 0
if a > 0:
    print("a est positif")
elif a < 0:
    print("a est negatif")
else:
    print("a est nul")
```

L'instruction while

```
a = 0
while a < 7:
    a = a + 1
    print(a)
```

L'instruction while

L'instruction while

```
n = 1
while n <= 5:
    print(n)
    n = n + 1
else:
    print("La boucle est terminee")
```

L'instruction for

Syntaxe à utiliser pour répéter un bloc d'instruction un nombre `n` de fois connu à l'avance :

L'instruction for... range(...)

```
for i in range(n):
    ...
    les differentes instructions a repeter
    ...
```

L'instruction for

Il y a ainsi trois possibilités pour utiliser l'instruction range, suivant le nombre de paramètres entiers saisis :

Premier cas

```
for i in range(n):
    ...
```

le compteur `i` va de `0` à `n-1` par pas de `1` : il y a donc bien `n` répétitions.

L'instruction for

Deuxième cas

```
for i in range(m, n):
    ...
```

le compteur `i` va de `m` à `n - 1` par pas de `1` : il y a donc `n - m` répétitions.
Ici, on a forcément `n>=m`.

L'instruction for

Troisième cas

```
for i in range(m, n, pas):
    ...
```

le compteur `i` va de `m` à `n - 1` par pas de `pas`. Le pas est un entier relatif non nul, il peut en particulier être négatif.

Comment choisir entre boucle for et boucle while

En général, si on connaît avant de démarrer la boucle le nombre d'itérations à exécuter, on choisit une boucle `for`.

Au contraire, si la décision d'arrêter la boucle ne peut se faire que par un test, on choisit une boucle `while`.

Fonctions natives

Les fonction prédéfinies sont des fonctions déjà créées et mises à notre disposition par Python.
Exemples :

- `eval()`
- `float`
- `help`
- `id()`
- `input()`
- `int()`
- `len()`
- `max()`
- `min()`
- `print()`
- `str()`
- `type()`

Fonctions définies par l'utilisateur

En plus des fonction prédéfinies, Python nous laisse la possibilité de définir nos propres fonctions.

Fonction

```
def nom_fonction(parametres):
    première instruction
    ....
    ....
    dernière instruction
```

Fonctions définies par l'utilisateur

Fonction sans paramètre

```
def hello():
    print("Hello")
```

Fonctions définies par l'utilisateur

Fonction avec un paramètre

```
def hello(prenom):
    print("Hello ", prenom)
```

Fonctions définies par l'utilisateur

Fonction avec l'instruction return

```
def carre(x):
    return x**2

>>> carre(3)
>>> 9
```