Partie I

Introduction à la programmation

Informatique 1
Introduction à la programmation

Mission 2 : INTRODUCTION

Types de données, Conversions, Boucles « for », Algorithmes

Kim Mens Siegfried Nijssen Charles Pecheur

Types de données

Toute valeur de donnée appartient à un **type**

```
>>> type(25)
<class 'int'>
```
Les **entiers**

```
>>> type(37.2)
<class 'float'>
```
Les **réels**

(réels en virgule flottante)

```
>>> type(True)
<class 'bool'>
```
Les **booléens**

```
>>> type("Bonjour")
<class 'str'>
```
Les **strings**

(chaînes de caractères)

```
>>> type(print)
<class 'builtin_function_or_method'>
```
Les **fonctions internes**

Type de données

`type(x)` retourne (une valeur représentant) le type de `x`

```
>>> x=5
>>> type(x)
<class 'int'>
```

Les **entiers**

```
>>> type(type(x))
<class 'type'>
```

Les **types de données**

```
>>> type(type(type(x)))
<class 'type'>
```

idem !

Type de données

Un type de données =

un **ensemble de valeurs** appartenant au type : le **domaine**

\+ un **ensemble d'opérations** applicables aux valeurs du type

Les **entiers** : `0, 1, 2, …, -1, -2, …`

`+, -, *, //, %, **`

Les **réels** : `0.0, 0.01, 6.02e-23, …`

`+, -, *, /, %, **`

Les **booléens** : `True,False`

`and, or, not`

`==, !=, <, >, <=, >=`

Et beaucoup d'autres…

Chaînes de caractères

Valeurs :

```
"Bonjour"
'Salut'
"Il dit 'non'"
"""Sur
plusieurs
lignes"""
```

concaténation
PAS addition !

Opérations

```
"2" + "2"                    "22"
"infor" + "matique"          "informatique"
"6" * 3                      "666"
```

répétition
PAS multiplication !

Affectation abrégée

Opération et affectation en même temps

`x += 1` "Ajouter 1 à x"

Equivaut à `x = x + 1`

Aussi pour −, *, **, /, //, %

```
x = 10 ; x -= 2
x = 10 ; x *= 5
x = 10 ; x //= 3
x = "10" ; x += "10"
```

```
x = 8
x = 50
x = 3
x = "1010"
```

Conversion de types

`int(x)` convertit `x` en valeur **entière**

```
int(3.14)          3
int(3.999)         3  (arrondit vers 0)
int("42")          42
int("15cm")        ValueError
```

`float(x)` convertit `x` en valeur **réelle**

```
float(42)          42.0
```

`str(x)` convertit `x` en chaîne de caractères

```
str(3.14)          "3.14"
```

Conversion implicite

Opérations arithmétiques : int => float

```
1 / 4          0.25        (int => float)
"1" + 4        TypeError
```

Egalité (entre tous types de valeurs)

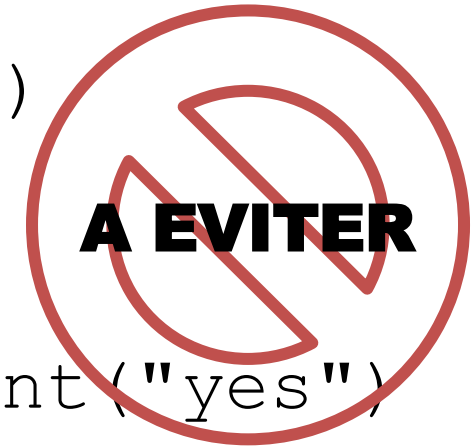
```
5 == 5.0       True        (int => float)
5 == "5"       False       (≠ types)
```

Comparaisons (entre valeurs numériques)

```
5 > 5.0        False       (int => float)
"5" > 0        TypeError
```

Conversion sur booléens

```
bool("True")          True
bool("False")         True      (≠ "")
bool("")              False     (= "")
bool("Hello")         True      (≠ "")
bool(0)               False
bool(1)               True      (≠ 0)
bool(42)              True      (≠ 0)
```




```
if 42:
    print("yes")          yes
```

Vérification de types

```
>>> type(3) == int
True

>>> isinstance(3,int)
True

>>> isinstance(3.14,float)
True

>>> isinstance(True,bool)
True

>>> isinstance("Bonjour",str)
True
```

Input

Entrer des données de l'utilisateur ?

```
>>> nom = input("Votre nom? ")
Votre nom? Sébastien
>>> nom
'Sébastien'
```

`input(msg)`

1. imprime `msg` à la console,
2. lit un texte à la console
3. retourne ce texte

Le résultat est toujours un **string**!

```
>>> age = input("Votre âge? ")
Votre âge? 10
>>> lustres = age//5
```

TypeError

Input

Entrer des données de l'utilisateur ?

```
>>> nom = input("Votre nom? ")
Votre nom? Sébastien
>>> nom
'Sébastien'
```

`input(msg)`

1. imprime `msg` à la console,
2. lit un texte à la console
3. retourne ce texte

Le résultat est toujours un **string**!

```
>>> age = int(input("Votre âge? "))
Votre âge? 10
>>> lustres = age//5
```

```
lustres = 2
```

Boucles for

```
for name in ["Charles", "Siegfried", "Kim"]:
    print("Cher", name, "vous êtes invité")
```

Boucles for

```
for name in ["Charles", "Siegfried", "Kim"]:
    print("Cher", name, "vous êtes invité")
```

(1) Séquence (éléments)

Boucles for

```
for name in ["Charles", "Siegfried", "Kim"]:
    print("Cher", name, "vous êtes invité")
```

(1) Séquence (éléments)

(3) Corps de la boucle

Boucles for

```
for name in ["Charles", "Siegfried", "Kim"]:
    print("Cher", name, "vous êtes invité")
```

```
Cher Charles vous êtes invité
Cher Siegfried vous êtes invité
Cher Kim vous êtes invité
```

(1) Séquence (éléments)

(3) Corps de la boucle

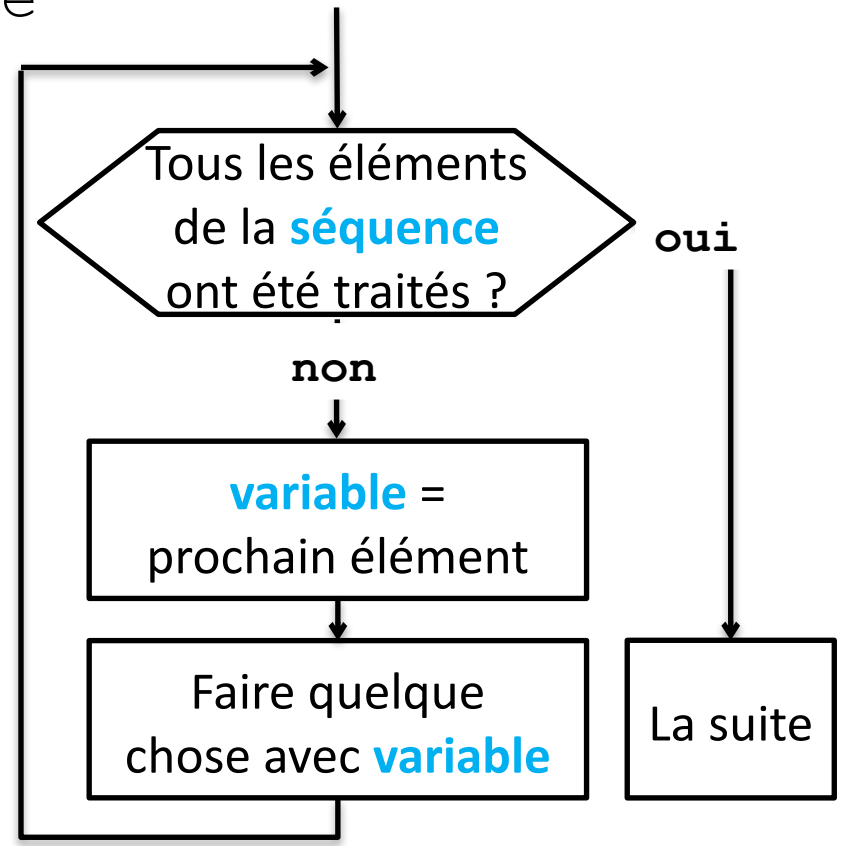
Une instruction de contrôle

`[a, b, c]` dénote une **liste** (expression)

Exécute le bloc **pour chaque valeur de la liste**

Boucle for

```
for variable in séquence:
    #
    # Faire quelque chose
    # avec variable
    #
#
# La suite
#
```



Boucles for

```
for angle in [0, 30, 45, 60, 90]:
    print(angle, "\t", angle/360)
```

```
0       0.0
30
0.08333333333333333
45      0.125
60
0.16666666666666666
90      0.25
```

```
for v in [1, 2.5, "hello", True]:
    print(v, "\t", type(v))
```

```
1       <class 'int'>
2.5     <class 'float'>
hello   <class 'str'>
True    <class 'bool'>
```

```
for x in [0, 1, 2, 3]:
    print(x, "\t", x*x)
for x in range(4):
    print(x, "\t", x*x)
n = int(input("n?"))
for x in range(n):
    print(x, "\t", x*x)
```

```
0       0
1       1
2       4
3       9
```

range

`range(n)`

retourne la séquence des **n** nombres de **0** à **n-1**

`range(n1, n2)`

retourne la séquence des **n2-n1** nombres de **n1** à **n2-1**

range

```
>> range(4)
range(0, 4)          ▭ un objet de type range
>>> list(range(4))
[0, 1, 2, 3]
>>> list(range(1, 5))
[1, 2, 3, 4]
>> range(4) == [0,1,2,3]
False    ▭ types différents
```

Algorithme

Un **algorithme** = une **procédure précise** permettant de **résoudre un problème**

Problème

Afficher le montant qui s'accumule sur un livret d'épargne pendant les 5 prochaines années si le livret offre un rendement donné x%

Algorithme

Problème

Afficher le montant qui s'accumule sur un livret d'épargne pendant les 5 prochaines années si le livret offre un rendement donné x%

Solution

Entrer le montant placé sur le livret

Entrer le taux d'intérêt

Pour chaque année:

Calculer l'intérêt à la fin de l'année

Afficher le montant total à la fin de l'année

Algorithme

Entrer le montant placé sur le livret

```
capital = float(input("Solde ? "))
```

Entrer le taux d'intérêt

```
taux = float(input("Taux ? "))
```

Pour chaque année:

```
duree = 5
for an in range(duree):
```

Calculer l'intérêt à la fin de l'année

```
    capital *= (1 + taux/100)
```

Afficher le montant total à la fin de l'année

```
    print(an+1, "\t", capital)
```

Algorithme

```
capital = float(input("Solde ? "))
taux = float(input("Taux ? "))
duree = 5
for an in range(duree):
        capital *= (1 + taux/100)
        print(an+1, "\t", capital)
```

```
Solde ? 1000
Taux ? 2.5
1       1025.0
2       1050.625
3       1076.890625
4       1103.812890625
5       1131.4082128906248
```