

Partie I

Introduction à la programmation

Informatique 1

Introduction à la programmation

Mission 2 : RESTRUCTURATION

Types de données, Opérations, Boucles « for »



Types de données

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

```
>>> type(42)
```

```
<class 'int'> : Les entiers
```

```
>>> type(37.2)
```

```
<class 'float'> : Les réels
```

```
>>> type(True)
```

```
<class 'bool'> : Les booléens
```

```
>>> type("Hello")
```

```
<class 'str'> : Les chaînes de caractères
```

```
>>> type(type("Hello"))
```

```
<class 'type'> : Les types de données
```

```
>>> type(type(type("Hello")))
```

```
<class 'type'>
```

Chaînes de caractères

```
>>> print('Salut')
```

```
Salut
```

```
>>> print("Bonjour")
```

```
Bonjour
```

```
>>> print("Hello')
```

```
SyntaxError
```

```
>>> print("'truc'")
```

```
'truc'
```

```
>>> print("gauche\"droite")
```

```
gauche"droite
```

```
>>> print("""Tic
```

```
Tac
```

```
Toe""")
```

```
Tic
```

```
Tac
```

```
Toe
```

```
>>> print("Tic\nTac\  
nToe")
```

```
Tic
```

```
Tac
```

```
Toe
```

Opérations sur chaînes

```
>>> print("5" + "3")
```

```
53
```

```
>>> print("5" * "3")
```

```
TypeError
```

```
>>> print("5" * 3)
```

```
555
```

```
>>> print(5 * "3")
```

```
33333
```

```
>>> print(("#" * 3 + "\n") * 3)
```

```
###
```

```
###
```

```
###
```

Conversion de types

<code>int(3.99)</code>	<code>3</code> (arrondi)
<code>int("42")</code>	<code>42</code>
<code>int("3.99")</code>	ERREUR
<code>int(True)</code>	<code>1</code>
<code>int(False)</code>	<code>0</code>
<code>float(42)</code>	<code>42.0</code>
<code>float("42")</code>	<code>42.0</code>
<code>float("3,99")</code>	ERREUR
<code>str(3.14)</code>	<code>'3.14'</code>
<code>str(1/3)</code>	<code>'0.33333333333333333333'</code>
<code>str(True)</code>	<code>'True'</code>

Conversion sur booléens

<code>bool("True")</code>	<code>True</code>	<code>(≠ "")</code>
<code>bool("False")</code>	<code>True</code>	<code>(≠ "")</code>
<code>bool("")</code>	<code>False</code>	<code>(= "")</code>
<code>bool("Hello")</code>	<code>True</code>	<code>(≠ "")</code>
<code>bool(0)</code>	<code>False</code>	
<code>bool(1)</code>	<code>True</code>	<code>(≠ 0)</code>
<code>bool(42)</code>	<code>True</code>	<code>(≠ 0)</code>

Conversion sur booléens

Les booléens sont compatibles avec les entiers !

```
True == 1
```

```
True
```

```
False == 0
```

```
True
```

```
10+True
```

```
11
```

```
42 and 64
```

```
64
```

```
if 42:
```

```
    print("yes")
```

```
yes
```



Input

```
>>> n = input("Combien ? ")
Combien ? 42
>>> n
'42'
>>> n // 2
TypeError
>>> int(n) // 2
21
>>> n * 2
'4242'
>>> print(2 ** int(input("bits:")))
bits:10
1024
```


Boucles for

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

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

- (1) Séquence (éléments)
- (2) Condition de traitement
- (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
```

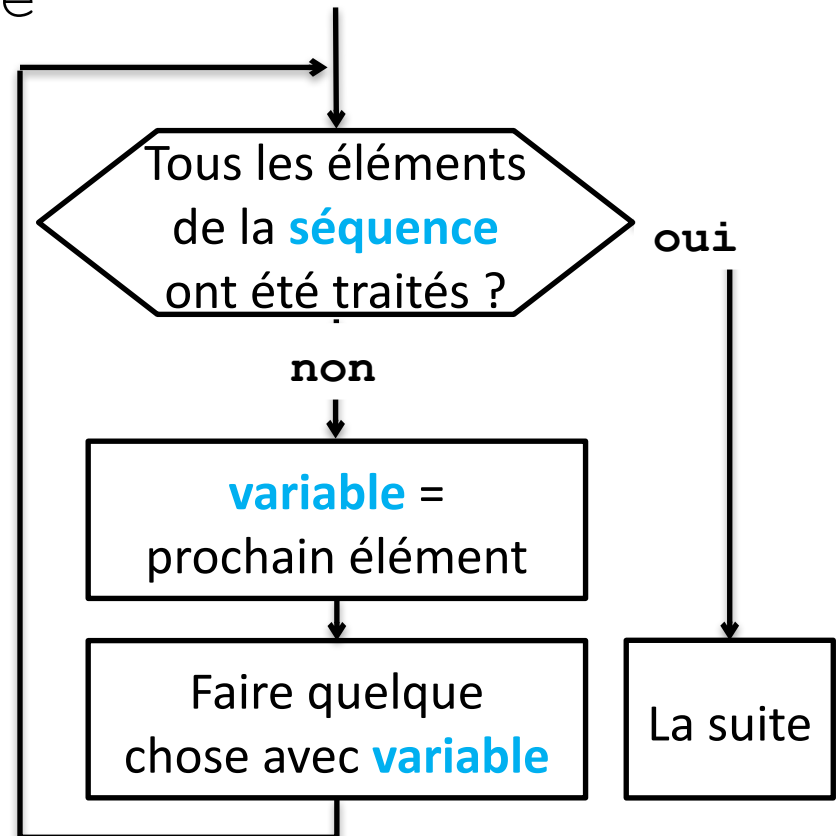
```
# avec variable
```

```
#
```

```
#
```

```
# La suite
```

```
#
```



Boucles for : séquences

(1) Séquence (éléments)

```
for n in [1/4, 1/2, 3/4]:  
    print (n)
```

```
0.25  
0.5  
0.75
```

(1) Séquence (intervalle)

```
for n in range(5):  
    print (n)
```

de 0 à 4

```
0  
1  
2  
3  
4
```

(1) Séquence (intervalle)

```
for n in range(5, 10):  
    print (n)
```

de 5 à 9

```
5  
6  
7  
8  
9
```

(1) Séquence (intervalle)

```
for n in range(0, 10, 2):  
    print (n)
```

de 0 à 9
par pas de 2

```
0  
2  
4  
6  
8
```

Boucles for : séquences

(1) Séquence (intervalle)

```
for n in range(10, 5, -1):  
    print(n)
```

de 10 à 6
à l'envers

```
10  
9  
8  
7  
6
```

de 10 à 6
à l'envers
par pas de 2

(1) Séquence (intervalle)

```
for n in range(10, 5, -2):  
    print(n)
```

```
10  
8  
6
```

Algorithme

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

Problème

Déterminer si un nombre est premier

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Comment déterminer si un *nombre* est premier ?

Vérifier s'il est divisible par 2, 3, 4, ..., *nombre-2*, *nombre-1*

Algorithme

Problème

Déterminer si un *nombre* est premier

Solution

Lire le *nombre*

Pour tous les *diviseurs* de 2 à *nombre*-1

Vérifier si le *nombre* est divisible par le *diviseur*

si oui, il n'est pas premier

il est premier

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Nombre premier

```
nombre = int(input("Votre nombre "))
est_premier = True
for diviseur in range(2, nombre):
    if nombre%diviseur == 0:
        est_premier = False
if est_premier:
    print(nombre, "est premier")
else:
    print(nombre, "n'est pas premier")
```

Peut-on aller plus vite ?

```
Votre nombre 25
25 n'est pas premier
Votre nombre 17
17 est premier
```

Sortie rapide

```
nombre = int(input("Votre nombre "))
est_premier = True
diviseur = 2
while diviseur < nombre and est_premier:
    if nombre%diviseur == 0:
        est_premier = False
        diviseur += 1
if est_premier:
    print(nombre, "est premier")
else:
    print(nombre, "n'est pas premier")
```



Sortie rapide

break

```
nombre = int(input("Votre nombre "))
est_premier = True
for diviseur in range(2, nombre):
    if nombre%diviseur == 0:
        est_premier = False
        break
if est_premier:
    print(nombre, "est premier")
else:
    print(nombre, "n'est pas premier")
```

Sort immédiatement de la boucle