Introduction à Python

Plan

CONTENUS	CAPACITÉS ATTENDUES	COMMENTAIRES
Variables et instructions élémentaires	• choisir ou déterminer le type d'une variable (entier, flottant ou chaîne de caractères) ; • concevoir et écrire des affectations à des variables ; • écrire une formule permettant un calcul combinant des variables.	On commence par consolider les notions de variables, de boucles et d'instructions conditionnelles introduites au cycle 4 en complétant la programmation par blocs par l'utilisation d'un langage de programmation textuel.
Boucle et itérateur, instruction conditionnelle	• programmer une instruction conditionnelle ; • programmer une boucle bornée ; • programmer une boucle non bornée.	On formalise les notions de boucle bornée (for) et de boucle non bornée (while) et on introduit la notion nouvelle de fonction dans un langage de programmation.
Notion de fonction	• programmer des fonctions simples, ayant un petit nombre d'arguments.	Il est intéressant de confronter les fonctions dans un langage de programmation avec les fonctions d'un tableur.

Objectifs

Objectifs

- Apprendre les bases du langage Python

Objectifs

- Apprendre les bases du langage Python
- S'approprier le vocabulaire de l'informatique

Objectifs

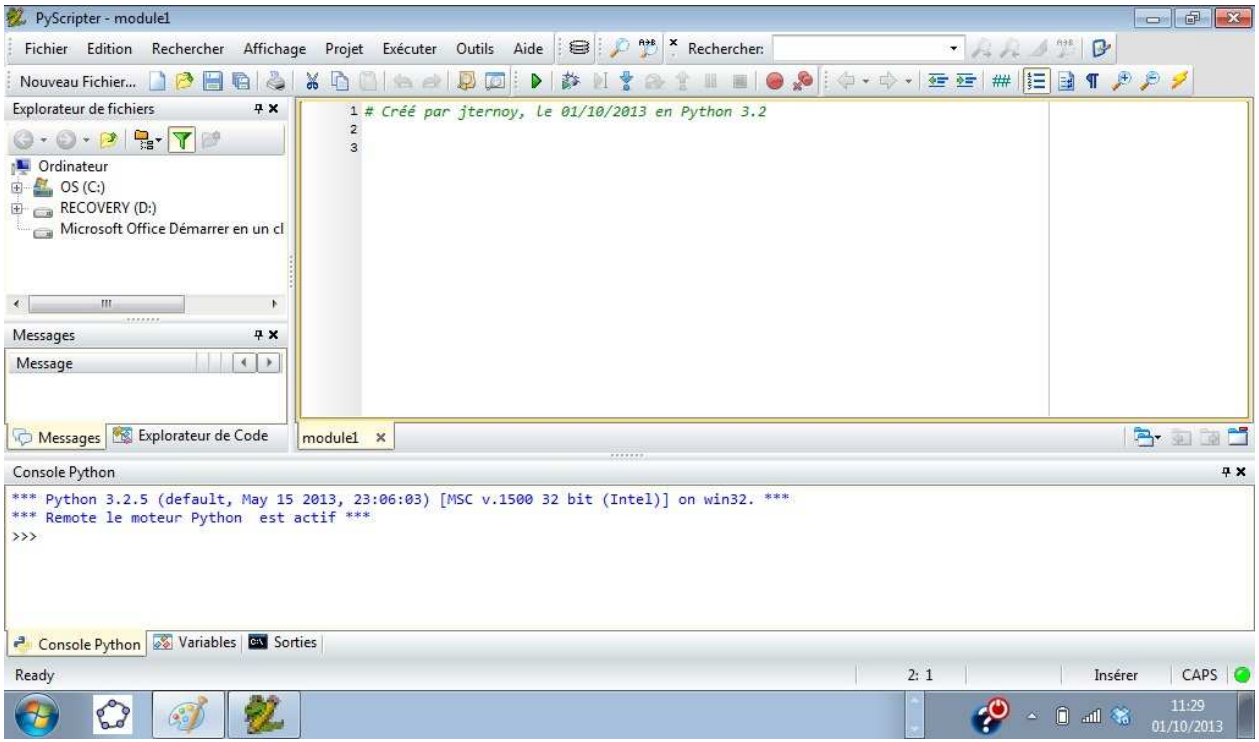
- Apprendre les bases du langage Python
- S'approprier le vocabulaire de l'informatique
- Connaître les bonnes pratiques de la programmation

Objectifs

- Apprendre les bases du langage Python
- S'approprier le vocabulaire de l'informatique
- Connaître les bonnes pratiques de la programmation

Plan

- Langage : Python 3.5 $\neq$ Python 2.7
- Éditeur : Edupython (ou autres...)
- 2 modes :
  - console/shell/terminal
  - script/fichier/programme



Plan

Plan

Variables

```
a = 2 #integer ou entier
a = 2.0 #float ou flottant
a = '2' #ou "2" ou '''2''' string ou chaine de caractère
a = 1+2*j #nombre complexe
a = true #booléen
a = [2,3,4] #liste(indice 0)
dictionnaires, tuples, sets, objets
```

Variables

```
a = 2 #integer ou entier
a = 2.0 #float ou flottant
a = '2' #ou "2" ou '''2''' string ou chaine de caractère
a = 1+2*j #nombre complexe
a = true #booléen
a = [2,3,4] #liste(indice 0)
dictionnaires, tuples, sets, objets
```

Particularités de Python

- Pas de déclaration ; typage dynamique
- Sensible à la casse
- Forcer le typage : int(2.0) ; int('2') ; str(2) ; ...
- Commentaires dans le code avec #

Plan

Opérations sur les nombres

```
>>> x = 45
45
>>> x + 2
47
>>> x = x + 3
48
>>> x / 3 #résultat est de type flottant alors que les
    entrées sont de type entier
16.0
>>>x**2 #puissance d'un nombre
2304
>>> 34//5; 34%5 #quotient et reste de la division de 34 par 5
6
4
```

Modules

```
>>> from math import *

>>> sqrt(5)   # de même exp(2); ln(3); cos(45)
2.236067977499 79


>>> from random import *

>>> random() # de même uniform(5,7); randint(3,10)
0.2494601147871 2649
```

Plan

Saisie

input("message") :

Instruction qui renvoie la chaine de caractère (str) saisie par l'utilisateur.

Saisie

input("message") :

Instruction qui renvoie la chaine de caractère (str) saisie par l'utilisateur.

```
>>> prenom = input("Entrez votre prénom : " )
Entrez votre prénom : Roger

>>> type(prenom)
<class 'str'>
```

Saisie

```
>>> age = input("Donnez votre age : " )
Donnez votre age : 5

>>> type(age)
<class 'str'>

>>> age+1
Traceback (most recent call last):
  File "<console>", line 1, in <module>
TypeError: Can't convert 'int' object to str implicitly

>>> int(age) + 1
6
```

Saisie

Erreurs classiques avec Input

- Input renvoie une valeur qu'il ne faut pas oublier d'affecter à une variable !
- 2 opérations à faire : Saisie et typage.
- age = int(input("Donnez votre age : "))

Affichage

print("message") ou print(variable) :

Instruction qui affiche une chaine de caractère ou le contenu de la variable

```
>>> x = 3

>>> y = 1000000000

>>> print(x)
3

>>> print(x,y) # affichage de plusieurs éléments en une seule
    instruction
3 1000000000

>>> print("L'an prochain, vous aurez",age+1,"ans.")
L'an prochain, vous aurez 7 ans.
```

Exemple

```
from math import *

Xa = float(input("Entrer l'abscisse du point A : "))
Ya = float(input("Entrer l'ordonnée du point A : "))
Xb = float(input("Entrer l'abscisse du point B : "))
Yb = float(input("Entrer l'ordonnée du point B : "))
print("La distance AB vaut ",sqrt((Xb-Xa)**2+(Yb-Ya)**2))
```

Plan

Les listes

```
>>> animaux = ['girafe','tigre','singe','souris']

>>> animaux[1]
'tigre'

>>> animaux[1:3]
['tigre', 'singe']

>>> animaux[2:]
['singe', 'souris']
```

Opération sur les listes

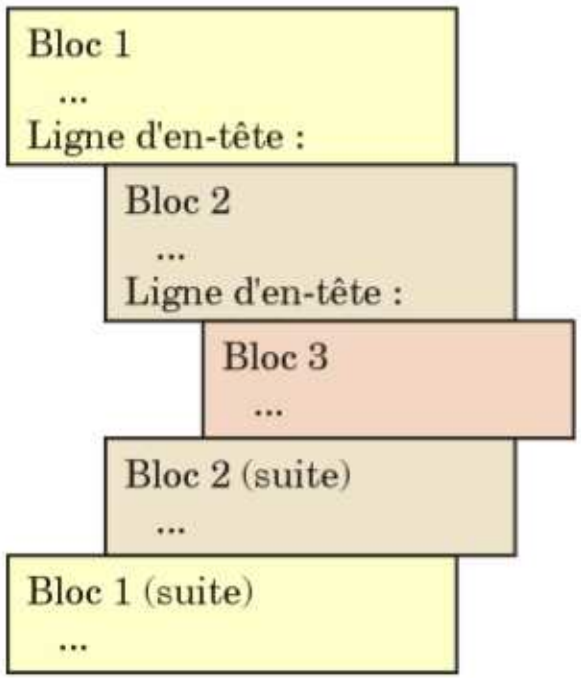
```
>>> l=[0,1,2,3,4,5]

>>> l = l+[6]

>>> print(l)
[0, 1, 2, 3, 4, 5, 6]

>>> l[5] = "cinq"

>>> print(l)
[0, 1, 2, 3, 4, "cinq", 6]
```

Plan

Plan

Principe général

- une ligne d'en-tête(*if*, *elif*, *else*, *for*, *while*, *def*, etc.) et se terminant par un double point.
- bloc d'instruction délimité par indentation.



Comparateurs

x == y	x est égal à y
x != y	x est différent de y
x > y	x est strictement supérieur à y
x < y	x est strictement inférieur à y
x <= y	x est inférieur ou égal à y
x >= y	x est supérieur ou égal à y
x < y < z	spécial Python
and	et
or	ou
not	non

Plan

if-then-else

```
if a == 0:
    print("L'équation est du premier degré.")
else:
    delta = b**2 - 4*a*c
    if delta > 0:
        print("L'équation possède deux solutions réelles.")
    else:
        if delta == 0:
            print("L'équation possède une solution réelle.")
        else:
            print("L'équation ne possède pas de solution
                réelle.")
```

elif

```
delta = b**2 - 4*a*c
if a == 0:
    print("L'équation est du premier degré.")
elif delta > 0:
    print("L'équation possède deux solutions réelles.")
elif delta == 0:
    print("L'équation possède une solution réelle.")
else:
    print("L'équation ne possède pas de solution réelle.")
```

Plan

while

```
>>> nombreEntre = 0;
>>> while (nombreEntre != 47) :
...     nombreEntre = int(input(" Tapez le nombre 47 ! ") )
...
...
Tapez le nombre 47 ! 5
Tapez le nombre 47 ! 9
Tapez le nombre 47 ! 47
```

Exemple : Suite de Syracuse

$$17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1$$

Exemple : Suite de Syracuse

$$17 - 52 - 26 - 13 - 40 - 20 - 10 - 5 - 16 - 8 - 4 - 2 - 1$$

Variable :	u, nombre entier
Initialisation :	Saisir u
Traitement :	Tant que $u \neq 1$ faire
	Si $u = 0[2]$ alors :
	Affecter à u la valeur $u/2$
	Sinon
	Affecter à u la valeur $3u + 1$
	Fin Si
	Afficher u
	Fin Tant que

```
u = int(input("u=?"))
while u!=1:
    if u%2==0:
        u = u/2
    else:
        u = 3*u+1
    print(u)
```

Plan

for

```
>>> for fruit in [''pomme'',''poire'',''kiwi'']:
...     print(fruit)
...
pomme
poire
kiwi
```

for

```
>>> for fruit in [''pomme'',''poire'',''kiwi'']:
...     print(fruit)
...
pomme
poire
kiwi
```

```
>>> for letter in "bac":
...     print(letter)
...
b
a
c
```

for

```
>>> for fruit in [''pomme'',''poire'',''kiwi'']:
...     print(fruit)
...
pomme
poire
kiwi
```

```
>>> for letter in "bac":
...     print(letter)
...
b
a
c
```

```
>>> for i in range(10):
...     print(i,end=";")
...
0;1;2;3;4;5;6;7;8;9;
```

```
>>> for i in range(15,20):
...     print(i,end=";")
...
...
15; 16; 17; 18; 19

>>> for i in range(0,1000,200):
...     print(i,end=";")
...
...
0; 200; 400; 600; 800

>>> for i in range(2,-2,-1):
...     print(i,end=";")
...
...
2; 1; 0; -1
```

Plan

fonctions

```
>>> def f(x):
...     "fonction carrée"
...     return(x**2)
...
...

>>> f(5)
25
```

fonctions

```
>>> def f(x):
...     "fonction carrée"
...     return(x**2)
...
...

>>> f(5)
25
```

Attention !

- Définition en début de programme ;
- Appel dans le corps du programme ;

fonctions à plusieurs paramètres

```
>>> def norme(x,y):
...     return(sqrt(x**2+y**2))
...
...

>>> norme(3,4)
5.0
```

fonctions-Portée des variables

```
def test1():
    p = 7
    print(p)

test1()
print(p)
```

```
def test2():
    p = 7
    print(p)

p = 8
test2()
print(p)
```

Quels résultats ?

fonctions-Portée des variables

```
def test1():
    p = 7
    print(p)

test1()
print(p)
```

```
def test2():
    p = 7
    print(p)

p = 8
test2()
print(p)
```

Quels résultats ?

- test1 : 7 et erreur variable non définie

fonctions-Portée des variables

```
def test1():
    p = 7
    print(p)

test1()
print(p)
```

```
def test2():
    p = 7
    print(p)

p = 8
test2()
print(p)
```

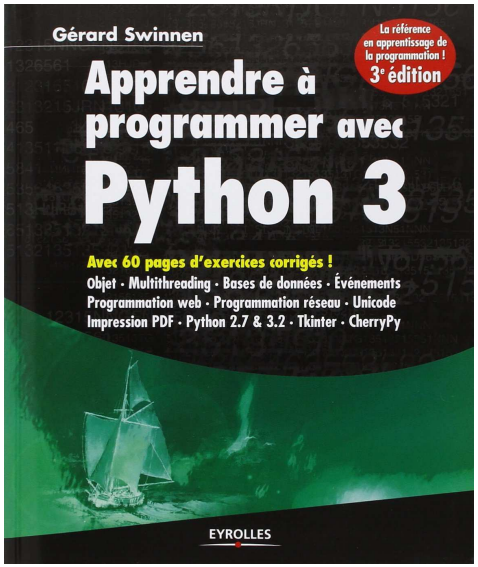
Quels résultats ?

- test1 : 7 et erreur variable non définie
- test2 : 7 et 8

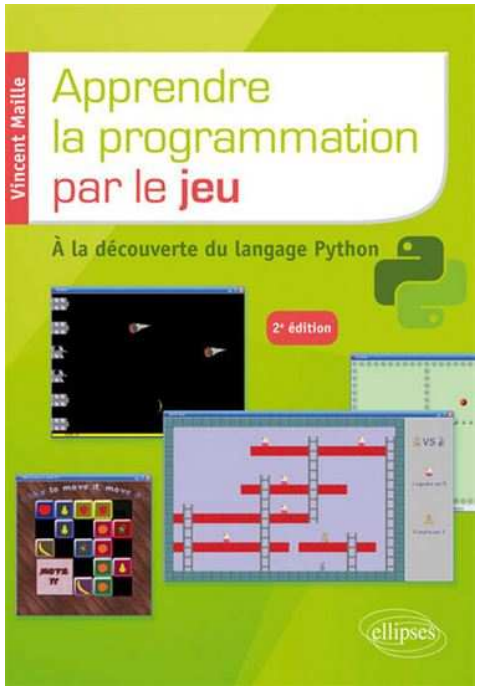
Plan

```
# Created on 29/06/2017
# Author : ASS

###### Les fonctions
def factorielle(n):
    """ Fonction factorielle """
    facto = 1
    for i in range(1, n+1):
        facto = facto * i
    return facto

###### Corps du programme
print("Ce programme affiche les n premieres factorielles")
fin = int(input("Combien de factorielles souhaitez-vous
    afficher ? "))
for i in range(1, fin+1):
    print("factorielle de",i," vaut ",factorielle(i))
###### fin du programme
```

Plan



Gerard Swimmen



Vincent Maille

Documentation pour l'enseignant

version 1.3 du 29 décembre 2015

Site d'Edupython