

# Atelier d'informatique

## Épisode I : Introduction

1<sup>er</sup> février 2017

« *Un programme informatique fait ce que vous lui avez dit de faire, pas ce que vous voulez qu'il fasse.* » — Troisième loi de Greer

- 1 L'informatique, c'est quoi ?
- 2 Espace de travail
- 3 Calculs élémentaires
- 4 Variables
- 5 Chaînes de caractères
  - Généralités
  - Opérations sur les chaînes
- 6 Premiers programmes

# Informatique : késako ?

Le mot *informatique* est la contraction des mots *information* et *automatique* : il s'agit donc de la science du traitement automatique de l'information.

# Informatique : késako ?

Le mot *informatique* est la contraction des mots *information* et *automatique* : il s'agit donc de la science du traitement automatique de l'information.

Un *ordinateur* est la concrétisation de cette notion, une machine qui traite automatiquement des informations données en entrée, selon un *programme informatique* qui dicte comment procéder.

# Comment Pythonner

- Démarrez sur votre bureau le programme **Pyzo**.

# Comment Pythonner

- Démarrez sur votre bureau le programme **Pyzo**.
- **Pyzo** est un environnement de développement pour le langage de programmation Python. Il inclut une **console** (ou *shell*, ou *interpréteur*), où sont entrées les instructions à exécuter immédiatement, et une zone où écrire des scripts.

# Comment Pythonner

- Démarrez sur votre bureau le programme **Pyzo**.
- **Pyzo** est un environnement de développement pour le langage de programmation Python. Il inclut une **console** (ou *shell*, ou *interpréteur*), où sont entrées les instructions à exécuter immédiatement, et une zone où écrire des scripts.

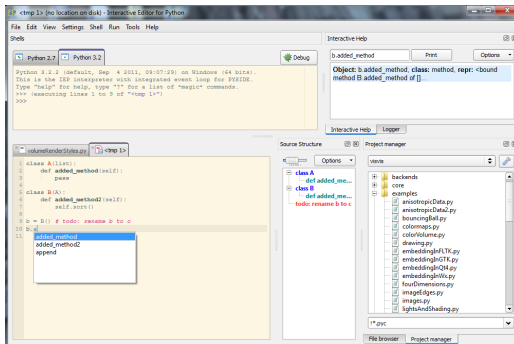


FIG.: Pyzo.

# Premiers pas

## Exercice 1

On va commencer par se familiariser avec les opérations arithmétiques de base que peut faire Python.

**Addition** Entrer  $2+2$  dans la console, appuyer sur Entrée pour exécuter votre saisie. Vérifier que ça fait 4.



# Premiers pas

## Exercice 1

On va commencer par se familiariser avec les opérations arithmétiques de base que peut faire Python.

**Addition** Entrer  $2+2$  dans la console, appuyer sur Entrée pour exécuter votre saisie. Vérifier que ça fait 4.

**Soustraction** Entrer  $4-3$ . Vérifier que l'on trouve 1.

# Premiers pas

## Exercice 1

On va commencer par se familiariser avec les opérations arithmétiques de base que peut faire Python.

**Addition** Entrer  $2+2$  dans la console, appuyer sur Entrée pour exécuter votre saisie. Vérifier que ça fait 4.

**Soustraction** Entrer  $4-3$ . Vérifier que l'on trouve 1.

**Produit** Taper  $2*3$ . Vérifier que l'on trouve 6.

# Premiers pas

## Exercice 1

On va commencer par se familiariser avec les opérations arithmétiques de base que peut faire Python.

**Addition** Entrer  $2+2$  dans la console, appuyer sur Entrée pour exécuter votre saisie. Vérifier que ça fait 4.

**Soustraction** Entrer  $4-3$ . Vérifier que l'on trouve 1.

**Produit** Taper  $2*3$ . Vérifier que l'on trouve 6.

**Division** Taper  $12/3$ . Vérifier que l'on trouve 4. Taper  $4/5$ . Vérifier que l'on trouve 0.8.

# Premiers pas

## Exercice 1

On va commencer par se familiariser avec les opérations arithmétiques de base que peut faire Python.

**Addition** Entrer  $2+2$  dans la console, appuyer sur Entrée pour exécuter votre saisie. Vérifier que ça fait 4.

**Soustraction** Entrer  $4-3$ . Vérifier que l'on trouve 1.

**Produit** Taper  $2*3$ . Vérifier que l'on trouve 6.

**Division** Taper  $12/3$ . Vérifier que l'on trouve 4. Taper  $4/5$ . Vérifier que l'on trouve 0.8.

**...de CM1** Taper  $13//4$ , et  $13\%4$ . Compléter la division posée suivante :

$$\begin{array}{r|l} 13 & 4 \\ \hline \dots & \dots \end{array}$$

Que remarquez-vous ?

# Premiers pas

## Exercice 1

On va commencer par se familiariser avec les opérations arithmétiques de base que peut faire Python.

**Addition** Entrer  $2+2$  dans la console, appuyer sur Entrée pour exécuter votre saisie. Vérifier que ça fait 4.

**Soustraction** Entrer  $4-3$ . Vérifier que l'on trouve 1.

**Produit** Taper  $2*3$ . Vérifier que l'on trouve 6.

**Division** Taper  $12/3$ . Vérifier que l'on trouve 4. Taper  $4/5$ . Vérifier que l'on trouve 0.8.

**...de CM1** Taper  $13//4$ , et  $13\%4$ . Compléter la division posée suivante :

$$\begin{array}{r|l} 13 & 4 \\ \hline \dots & \dots \end{array}$$

Que remarquez-vous ?

**Puissances** Taper  $2**3$ . Quel est le résultat ? Et celui de  $3**2$  ?

# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

Le fait d'affecter une valeur à une variable s'appelle une « *affectation* ».

En Python, on fait ça avec une expression de la forme `x=...`



# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

Le fait d'affecter une valeur à une variable s'appelle une « *affectation* ».

En Python, on fait ça avec une expression de la forme `x=...`

On peut également réaffecter une variable en recyclant simplement son nom.

# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

Le fait d'affecter une valeur à une variable s'appelle une « *affectation* ».

En Python, on fait ça avec une expression de la forme `x=...`

On peut également réaffecter une variable en recyclant simplement son nom.

Pour afficher la valeur d'une variable, on peut demander à Python de l'évaluer en tapant son nom.

# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

Le fait d'affecter une valeur à une variable s'appelle une « *affectation* ».

En Python, on fait ça avec une expression de la forme `x=...`

On peut également réaffecter une variable en recyclant simplement son nom.

Pour afficher la valeur d'une variable, on peut demander à Python de l'évaluer en tapant son nom.

## Exercice 2

- Créer une variable `x` qui a la valeur 3. Est-ce qu'on peut écrire `3=x` ?

# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

Le fait d'affecter une valeur à une variable s'appelle une « *affectation* ».

En Python, on fait ça avec une expression de la forme `x=...`

On peut également réaffecter une variable en recyclant simplement son nom.

Pour afficher la valeur d'une variable, on peut demander à Python de l'évaluer en tapant son nom.

## Exercice 2

- Créer une variable `x` qui a la valeur 3. Est-ce qu'on peut écrire `3=x` ?
- Créer une variable `y` qui a la valeur `7x`. Modifier la valeur de `x`. Cela modifie-t-il la valeur de `y` ?

# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

Le fait d'affecter une valeur à une variable s'appelle une « *affectation* ».

En Python, on fait ça avec une expression de la forme `x=...`

On peut également réaffecter une variable en recyclant simplement son nom.

Pour afficher la valeur d'une variable, on peut demander à Python de l'évaluer en tapant son nom.

## Exercice 2

- Créer une variable `x` qui a la valeur 3. Est-ce qu'on peut écrire `3=x` ?
- Créer une variable `y` qui a la valeur `7x`. Modifier la valeur de `x`. Cela modifie-t-il la valeur de `y` ?
- Rajouter 1 à `x` en la réaffectant, vérifier en évaluant. Même question avec 0.5.

# Variables

Mais Python est bien plus puissant qu'une simple *Casio collège*...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

Le fait d'affecter une valeur à une variable s'appelle une « *affectation* ».

En Python, on fait ça avec une expression de la forme `x=...`

On peut également réaffecter une variable en recyclant simplement son nom.

Pour afficher la valeur d'une variable, on peut demander à Python de l'évaluer en tapant son nom.

## Exercice 2

- Créer une variable `x` qui a la valeur 3. Est-ce qu'on peut écrire `3=x` ?
- Créer une variable `y` qui a la valeur `7x`. Modifier la valeur de `x`. Cela modifie-t-il la valeur de `y` ?
- Rajouter 1 à `x` en la réaffectant, vérifier en évaluant. Même question avec 0.5.
- Que font `x+=1`, `x-=1`, `x*=2` ou encore `x/=2` ?

# Variables

Mais Python est bien plus puissant qu'une simple *Casio* collège...

Une *variable* est la donnée d'un emplacement mémoire où une valeur est stockée, et d'un nom.

Le fait d'affecter une valeur à une variable s'appelle une « *affectation* ».

En Python, on fait ça avec une expression de la forme `x=...`

On peut également réaffecter une variable en recyclant simplement son nom.

Pour afficher la valeur d'une variable, on peut demander à Python de l'évaluer en tapant son nom.

## Exercice 2

- Créer une variable `x` qui a la valeur 3. Est-ce qu'on peut écrire `3=x` ?
- Créer une variable `y` qui a la valeur `7x`. Modifier la valeur de `x`. Cela modifie-t-il la valeur de `y` ?
- Rajouter 1 à `x` en la réaffectant, vérifier en évaluant. Même question avec 0.5.
- Que font `x+=1`, `x-=1`, `x*=2` ou encore `x/=2` ?
- Entrer l'instruction `x,y=y,x`. Quelles sont alors les valeurs de `x` et de `y` ?

**Remarque** Dans un langage de programmation un peu plus *kasher*, on déclare et on affecte séparément une variable. Mais Python permet de déclarer une variable directement par affectation.



**Remarque** Dans un langage de programmation un peu plus *kasher*, on déclare et on affecte séparément une variable. Mais Python permet de déclarer une variable directement par affectation.

### À propos des types

La fonction `type`, appliquée à une variable `x` via l'instruction `type(x)`, renvoie le type de la variable `x`. Essayez sur plusieurs valeurs : `type(0)`, `type(0.5)`, `type(type)`, et `type(0.)`.

**Remarque** Dans un langage de programmation un peu plus *kasher*, on déclare et on affecte séparément une variable. Mais Python permet de déclarer une variable directement par affectation.

## À propos des types

La fonction `type`, appliquée à une variable `x` via l'instruction `type(x)`, renvoie le type de la variable `x`. Essayez sur plusieurs valeurs : `type(0)`, `type(0.5)`, `type(type)`, et `type(0.)`.

Les types `int` et `float` servent respectivement à représenter les nombres entiers et les nombres réels à virgule dans Python. On peut faire les opérations arithmétiques vues plus haut avec elles, même quand les variables en jeu sont d'un type et de l'autre.

# Chaînes de caractères

## Généralités

Un nouveau type de variable important : le type `str`, pour *string* ou *chaîne de caractères*, en français. On peut les définir explicitement en incluant son message entre guillemets simples ( `'` ) ou doubles ( `"` ).

# Chaînes de caractères

## Généralités

Un nouveau type de variable important : le type `str`, pour *string* ou *chaîne de caractères*, en français. On peut les définir explicitement en incluant son message entre guillemets simples ( `'` ) ou doubles ( `"` ).  
Il faut faire attention : quand on démarre par une guillemet d'un type, l'apparition d'une autre guillemet du même type clôt la chaîne.

# Chaînes de caractères

## Généralités

Un nouveau type de variable important : le type `str`, pour *string* ou *chaîne de caractères*, en français. On peut les définir explicitement en incluant son message entre guillemets simples ( `'` ) ou doubles ( `"` ). Il faut faire attention : quand on démarre par une guillemet d'un type, l'apparition d'une autre guillemet du même type clôt la chaîne.

### Exercice 3

- Définir une variable `Bonjour` prenant la valeur 0.

# Chaînes de caractères

## Généralités

Un nouveau type de variable important : le type `str`, pour *string* ou *chaîne de caractères*, en français. On peut les définir explicitement en incluant son message entre guillemets simples ( `'` ) ou doubles ( `"` ). Il faut faire attention : quand on démarre par une guillemet d'un type, l'apparition d'une autre guillemet du même type clôt la chaîne.

### Exercice 3

- Définir une variable `Bonjour` prenant la valeur 0.
- Que fait `print(Bonjour)` ? Et les instructions `print("Bonjour")` et `print('Bonjour')` ?

# Chaînes de caractères

## Généralités

Un nouveau type de variable important : le type `str`, pour *string* ou *chaîne de caractères*, en français. On peut les définir explicitement en incluant son message entre guillemets simples ( `'` ) ou doubles ( `"` ). Il faut faire attention : quand on démarre par une guillemet d'un type, l'apparition d'une autre guillemet du même type clôt la chaîne.

### Exercice 3

- Définir une variable `Bonjour` prenant la valeur 0.
- Que fait `print(Bonjour)` ? Et les instructions `print("Bonjour")` et `print('Bonjour')` ?
- Comparer `type(Bonjour)`, `type('Bonjour')` et `type("Bonjour")`.

# Chaînes de caractères

## Généralités

Un nouveau type de variable important : le type `str`, pour *string* ou *chaîne de caractères*, en français. On peut les définir explicitement en incluant son message entre guillemets simples ( `'` ) ou doubles ( `"` ). Il faut faire attention : quand on démarre par une guillemet d'un type, l'apparition d'une autre guillemet du même type clôt la chaîne.

### Exercice 3

- Définir une variable `Bonjour` prenant la valeur 0.
- Que fait `print(Bonjour)` ? Et les instructions `print("Bonjour")` et `print('Bonjour')` ?
- Comparer `type(Bonjour)`, `type('Bonjour')` et `type("Bonjour")`.
- Évaluer `len("Bonjour")`. À quoi cela correspond-t-il ?



# Chaînes de caractères

## Généralités

Un nouveau type de variable important : le type `str`, pour *string* ou *chaîne de caractères*, en français. On peut les définir explicitement en incluant son message entre guillemets simples (') ou doubles ("). Il faut faire attention : quand on démarre par une guillemet d'un type, l'apparition d'une autre guillemet du même type clôt la chaîne.

### Exercice 3

- Définir une variable `Bonjour` prenant la valeur 0.
- Que fait `print(Bonjour)` ? Et les instructions `print("Bonjour")` et `print('Bonjour')` ?
- Comparer `type(Bonjour)`, `type('Bonjour')` et `type("Bonjour")`.
- Évaluer `len("Bonjour")`. À quoi cela correspond-t-il ?
- Afficher `J'aime la tartiflette` et Il dit: `"Bonjour!"`. Attention à utiliser des types de guillemets différents !

# Chaînes de caractères

## Opérations

### Exercice 4

- Que se passe-t-il quand on affiche une chaîne contenant `\n` ?

# Chaînes de caractères

## Opérations

### Exercice 4

- Que se passe-t-il quand on affiche une chaîne contenant `\n` ?
- Définir deux chaînes de caractères `x` et `y`: que fait `print(x, y)` ?  
Est-ce que ça marche aussi si `x` est une variable de type `int` ?

# Chaînes de caractères

## Opérations

### Exercice 4

- Que se passe-t-il quand on affiche une chaîne contenant `\n` ?
- Définir deux chaînes de caractères `x` et `y`: que fait `print(x, y)` ?  
Est-ce que ça marche aussi si `x` est une variable de type `int` ?
- Évaluer la valeur de `x+y`, l'afficher via `print`.

# Chaînes de caractères

## Opérations

### Exercice 4

- Que se passe-t-il quand on affiche une chaîne contenant `\n` ?
- Définir deux chaînes de caractères `x` et `y`: que fait `print(x, y)` ? Est-ce que ça marche aussi si `x` est une variable de type `int` ?
- Évaluer la valeur de `x+y`, l'afficher via `print`.
- Afficher « 1/100 est petit » en remplaçant 1/100 par sa valeur. On utilisera `"{} est petit".format()` où `x` est la valeur voulue.

# Premiers programmes

On va maintenant basculer sur la zone d'écriture de scripts. Un programme est une succession d'instructions qui sont effectuées lorsqu'il est exécuté.

# Premiers programmes

On va maintenant basculer sur la zone d'écriture de scripts. Un programme est une succession d'instructions qui sont effectuées lorsqu'il est exécuté.

## Exercice 5

- Écrire un programme qui stocke la chaîne « Bonjour » dans une variable `x`, puis l'affiche, puis affiche « Bonjour Bonjour ».

# Premiers programmes

On va maintenant basculer sur la zone d'écriture de scripts. Un programme est une succession d'instructions qui sont effectuées lorsqu'il est exécuté.

## Exercice 5

- Écrire un programme qui stocke la chaîne « Bonjour » dans une variable `x`, puis l'affiche, puis affiche « Bonjour Bonjour ».
- Écrire un programme qui demande à l'utilisateur d'entrer un texte, affiche « Vous avez entré: » suivi du texte entré. On utilisera la fonction `input`:

```
texte = input()
```

demande, à son exécution, une chaîne de caractère à l'utilisateur, puis la stocke dans la variable `texte`.



# Premiers programmes

On va maintenant basculer sur la zone d'écriture de scripts. Un programme est une succession d'instructions qui sont effectuées lorsqu'il est exécuté.

## Exercice 5

- Écrire un programme qui stocke la chaîne « Bonjour » dans une variable `x`, puis l'affiche, puis affiche « Bonjour Bonjour ».
- Écrire un programme qui demande à l'utilisateur d'entrer un texte, affiche « Vous avez entré: » suivi du texte entré. On utilisera la fonction `input`:

```
texte = input()
```

demande, à son exécution, une chaîne de caractère à l'utilisateur, puis la stocke dans la variable `texte`.

- Écrire un programme qui demande à l'utilisateur un nombre, le stocke dans une variable `x`, puis affiche «  $f(x) =$  » suivi de la valeur de  $\frac{1}{1+x^2}$ . On pourra utiliser `input`, convertir son entrée (initialement de type `str`) en un entier via le constructeur `int`.