

# INFO-H-100 - Informatique

Séance d'exercices 1

Introduction à Python

Valeurs, variables et expressions

Université libre de Bruxelles  
École polytechnique de Bruxelles

2018-2019

# Organisation

Travaux pratiques :

- 12 séances,
- sur papier et sur machine ([consulter votre horaire sur Gehol](#)).

Site web :

- <http://uv.ulb.ac.be>
- Enoncés d'exercices et anciens examens, certaines corrections, projets et horaires.

Assistants :

- |                          |  |
|--------------------------|--|
| • Jean Rosenfeld         | <a href="mailto:jrosenfe@ulb.ac.be">jrosenfe@ulb.ac.be</a> |
| • Jean-Philippe Hubinont | <a href="mailto:jhubinon@ulb.ac.be">jhubinon@ulb.ac.be</a> |
| • Alain Silovy           | <a href="mailto:asilovy@ulb.ac.be">asilovy@ulb.ac.be</a>   |
| • Ken Hasselmann         | <a href="mailto:khasselm@ulb.ac.be">khasselm@ulb.ac.be</a> |

# Guidances

Les [élèves assistants](#) sont disponibles pour vous certains midis de 12h30 à 13h30 au local [UB4.130](#) (Salle Socrate). Consultez le site web pour savoir quand ils sont disponibles.

Ils sont là pour [répondre à vos questions](#) sur :

- les machines des salles,
- les exercices de programmation,
- les projets,
- l'installation de Python chez vous,
- ...

# Evaluation

Un projet.

Examen de janvier (théorie et pratique).

Pondération en première session :

- 5% présence (mini-interrogations),
- 15% projets,
- 80% examen de janvier.

Pondération en seconde session :

- À confirmer...

# Mini-interrogations

- Les interros font actes de présence.
- Début à l'heure-10 et fin à l'heure-15.
- Un retard constitue une absence !
- Format : 1 question sur la séance précédente et 1 question sur la séance du jour.

Pourquoi apprendre  
l'informatique ?

# A-t'on besoin d'informaticiens ?

- Les ingénieurs informaticiens ou informaticiens ont un des taux d'embauche les plus élevés (plus de 90%) en deans les 6 mois après leur sortie de l'université
- Les informaticiens ont le salaire de départ moyen le plus élevé de toutes les études universitaires (Forbes)
- Automatisation : Tous les domaines de l'industrie ont de plus en plus besoin de personnes qualifiées en programmation (physique, chimie, mathématique, biologie, géologie, génie électrique, génie civil)
- 4ème révolution industrielle : Le digital et la technologie

## 4eme révolution industrielle : Les informaticiens sont partout

- **Relation client et consommateur** : design des services (sites web, le e-commerce) et la gestion des contenus digitaux (clouds, réseaux sociaux, systèmes de recommandation, sécuriser toutes ces données)
- **Fonctionnement interne d'une entreprise** : automatisation, digitalisation. Toutes les entreprises vont devoir changer radicalement leur organisation.
- **Internet des objets** : Les objets connectés "intelligents" apparaissent sur le marché de façon exponentielle. Un "smart bed" qui réveille une personne en fonction de ses cycles de sommeil ; Un dosage "intelligent" et automatique d'insuline pour les diabétiques ; etc.



# Innovation et Progrès

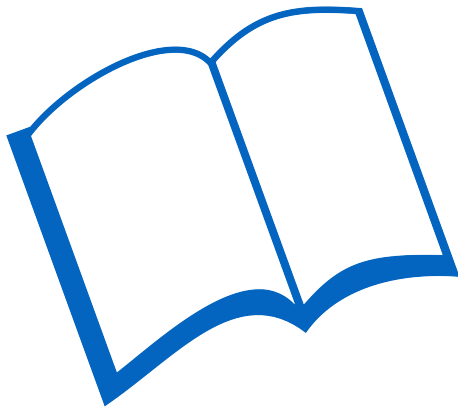
- **L'innovation de nos jours** : N'importe qui possédant un ordinateur peut créer et innover (Amazon, Facebook, etc.). Nous sommes à l'ère du service et de l'amélioration du bien-être des consommateurs
- **Renouveau des métiers** : Beaucoup de métiers vont disparaître ou être très fortement modifiés dans les 15 prochaines années
- **Objectifs d'apprentissage** : vous former à apprendre par vous même ; vous donner les outils pour surfer sur cette nouvelle révolution  
(<https://openclassrooms.com/courses/apprenez-a-programmer-en-python>)

# C'est quoi un bon informaticien ?

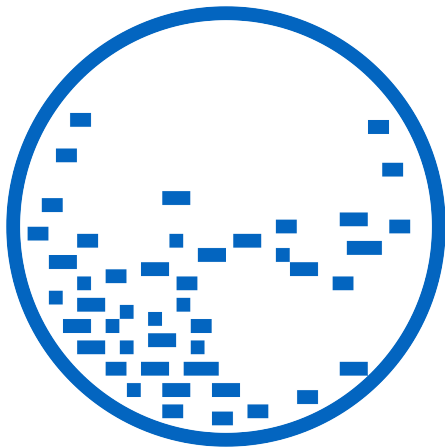
- Oubliez l'image de l'informaticien seul devant son ordinateur toute la journée. L'informatique est inévitable pour tout le monde et le profil le plus recherché est une personne qualifiée en programmation mais douée de talents de communication et relationnels.

Geek is the new cool

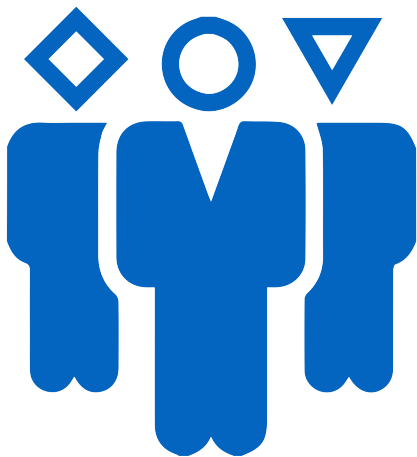
Etudiez et entrainez vous



Voyagez et apprenez



Collaborez



# Introduction à Python

Valeurs, variables et expressions

# Valeurs et types

Une **valeur** est un des éléments de base d'un programme, comme un nombre, une lettre ou une chaîne de caractère.

Toute valeur possède un **type** :

- 'Hello' est une chaîne de caractères (**string**) identifiable grâce aux apostrophes : `str`
- 1 et 2 sont des entiers (**integer**) : `int`
- 1.4 et 2.0 sont des réels (**floating-point number**) : `float`

La **fonction** `type` donne le type d'une valeur :

```
>>> type(2.0)
<type 'float'>
```

# Variables et assignation

Une **variable** est un nom qui permet de faire référence à une valeur.

L'**assignation** est l'instruction qui permet d'associer une valeur à une variable. Si la variable n'existe pas dans le contexte, elle est créée.

Syntaxe : `variable = valeur`

```
>>> i = 4
>>> message = '2 * 2 ='
>>> i
4
>>> print(i)
4
>>> print(message, i)
2 * 2 = 4
>>> i = 5
>>> i
5
```



# Variables

Le **type d'une variable** est le type de la valeur associée à cette variable.

```
| >>> type(i)  
| <type 'int'>
```

Le **nom d'une variable** ne peut contenir que des lettres, des chiffres et le caractère \_ (underscore). Il doit commencer par une lettre ou un underscore.

Veillez à donner à vos variables des **noms qui ont du sens** :  
average, sum, message, ...

Attention à la casse : totalPrice est différent de  
totalprice.

# Opérateurs

Les **opérateurs** suivants sont présents dans le langage :

- $+$ ,  $-$ ,  $*$ ,  $/$
- $**$  : exposant ( $5 ** 2$ )
- $\%$  : modulo, reste de la division entière ( $7 \% 2$ )
- $//$  : division entière

La **priorité des opérateurs** est la même qu'en arithmétique.

Les parenthèses peuvent également être utilisées.

# Opérateurs - division

Attention à la **division** !

L'opérateur `/` effectuera une **division en virgule flottante** même si les deux nombres sont des **entiers** (int).

L'opérateur `//` permet d'effectuer une **division entière**.

Exemple :

```
>>> 7 / 2
3.5          #type float
>>> 7.0 / 2
3.5          #type float
>>> 7 // 2
3            #type int
>>> 7.0 // 2
3.0          #type float
```

# Expressions

Une **expression** dénote une **valeur**. Elle est constituée d'opérateurs et d'opérandes.

Les opérandes peuvent être des valeurs, des variables ou des expressions.

```
>>> x = 3
>>> 17 + x
20
>>> x = (x - 1.0) ** 4
>>> x
16.0
>>> message = 'Hello' + ' ' + 'World'
>>> print(message)
Hello World
>>> 'ha' * 3
'hahaha'
```

# Affichage et chaînes de caractères

L'instruction `print` affiche le résultat d'une expression.

```
>>> print('spam' * 3)
spamspamspam
>>> print(2**1, 2**2) #sur une ligne
2 4
```

Caractères spéciaux et échappement :

```
>>> print('Spam \t Spam') #tabulation
Spam    Spam
>>> print('SPAM \n SPAM') #passage de ligne
SPAM
  SPAM
>>> print('\\', '\\', "'", '"')
\ ' " '
```

# Python 3.x

La [version de Python](#) enseignée dans ce cours est la [3.x](#).

*Il existe une version 2 de Python qui n'est pas entièrement compatible avec la version 3 que nous utilisons dans ce cours.*

Veillez donc à utiliser les outils [Python 3.x](#) dans les salles ainsi que chez vous pour les [exercices](#) et les [projets](#).

Pour lancer l'environnement de développement Jet Brains pycharm avec Python 3.x dans les salles, ouvrez un terminal et tapez

- `python.sh`

# Fonction input

La fonction `input` affiche un message à l'utilisateur, attend que celui-ci rentre des caractères terminés par ENTER et renvoie le texte correspondant.

```
>>> nom = input("Entrez votre nom : ")
Entrez votre nom : Pierre
>>> print(nom)
Pierre
>>> type(nom)
<type 'str'>
```

# Mode interactif et mode script

Dans les premiers exercices nous utilisons le **mode interactif** de Python (Python console), pratique pour tester de petites choses.

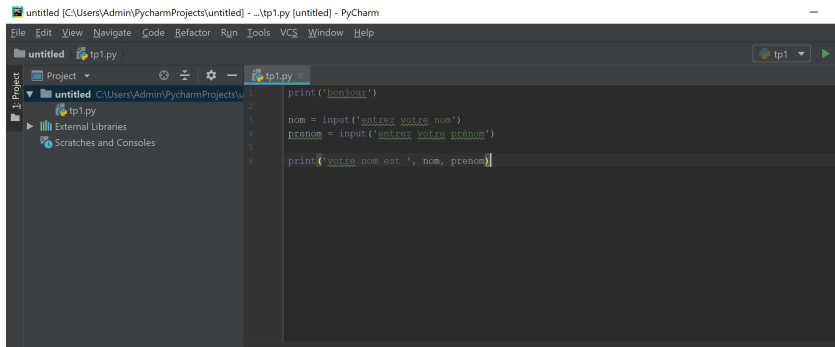
Pour faire un **programme** (ou **script**) réutilisable, il faut enregistrer les instructions dans un **fichier** (ici `cercle.py`) et l'exécuter.

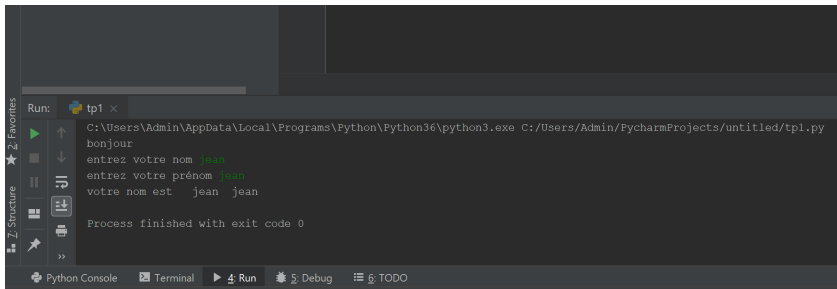
```
import math
rayon = input("Entrez le rayon en cm : ")
rayon = float(rayon)
aire = math.pi * (rayon**2)
print("Aire = " + str(aire) + " cm carres")
```



# Créer et exécuter un script avec Pycharm

- Créer ou ouvrir un projet pycharm
- Créer un nouveau fichier python (clic droit, New/Python File). Donner un nom au fichier. Enregistrer.
- Écrire votre programme.
- Exécuter le script (Triangle 'play' vert : Run).
- Le programme s'exécute dans la fenêtre de l'interpréteur : Run.





```
Run: tp1 x
C:\Users\Admin\AppData\Local\Programs\Python\Python36\python3.exe C:/Users/Admin/PycharmProjects/untitled/tp1.py
bonjour
entrez votre nom jean
entrez votre prénom jean
votre nom est  jean jean

Process finished with exit code 0
```

Python Console | Terminal | 4 Run | 5 Debug | 6 TODO

## Démonstration

# Exercices

1 à 12.

Tips :

```
>>> import math    #bibliotheque math
>>> math.pi
3.141592653589793
>>> math.e
2.718281828459045
```

```
>>> str(12)
'12'
>>> int(23.7)
23
>>> round(23.7)
24.0
>>> int('123')
123
>>> float(2)
2.0
```

# Connection au machines et UPyLab

Si vous n'avez pas de netid actif, vous pouvez accéder aux ordinateurs et à UpyLaB via :

- login : INFOH100
- password : INFOH100

Démo de UpyLaB en fin de TP : `http:\\upylab.ulb.ac.be`

# Prochain TP

Script et fonctions :

- input
- mode interactif et mode script
- fonctions

Voir les slides et énoncé du TP suivant !