INFO-H-100 - Informatique

Séance d'exercices 1
Introduction à Python
Valeurs, variables et expressions

Université libre de Bruxelles
École polytechnique de Bruxelles

2013-2014

Organisation

Travaux pratiques :

- 18 séances,
- Sur papier et sur machine (consulter votre horaire sur le site web des TPs).

Site web :

- `http://cs.ulb.ac.be/public/teaching/infoh100`
- Enoncés, certaines corrections, projets et horaires.

Assistants :

- Alain Silovy - `asilovy@ulb.ac.be`
- Anh Vu Doan - `ndoan1@ulb.ac.be`
- Michaël Waumans - `mwaumans@ulb.ac.be`
- Dimitri Van Assche - `dvassche@ulb.ac.be`
- Stefan Eppe - `stefan.eppe@ulb.ac.be`

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

Deux projets (un par semestre).

Interrogation de janvier (théorie et pratique).

Examen de juin (théorie et pratique).

Pondération en première session :

- 20% projets,
- 20% interrogation de janvier,
- 60% examen de juin.

Pondération en seconde session :

- 20% projets,
- 80% examen de septembre.

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 3.2

La version de Python enseignée dans ce cours est la 3.2.

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 3.2 dans les salles ainsi que chez vous pour les exercices et les projets.

Pour lancer l'environnement de développement IDLE Python 3.2 dans les salles, ouvrez un terminal et tapez

- `idle-python3.2`

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

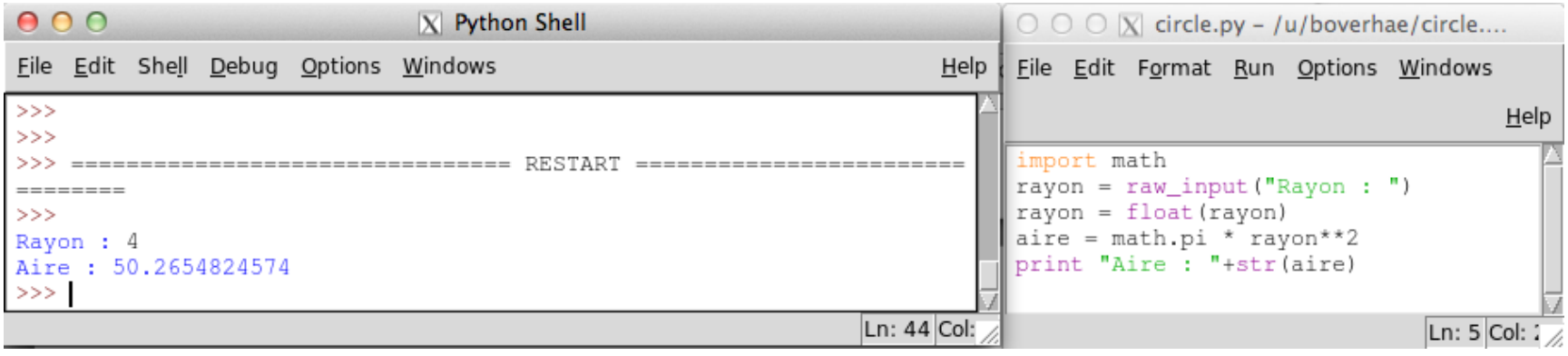
Dans les premiers exercices nous utilisons le mode interactif de Python, pratique pour tester de petites choses.

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 IDLE

- Ouvrir une nouvelle fenêtre (File, New Window).
- Écrire votre programme.
- Enregistrer cette fenêtre dans un fichier avec l'extension `.py` (File, Save).
- Exécuter le script (F5 ou Run, Python Shell).
- Le programme s'exécute dans la fenêtre de l'interpréteur.



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
```