

INFO-H-100 - Informatique

Séance d'exercices 2
Introduction à Python
Scripts et fonctions

Université Libre de Bruxelles
Faculté des Sciences Appliquées

2011-2012

Fonction `raw_input`

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

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

Mode interactif et mode script

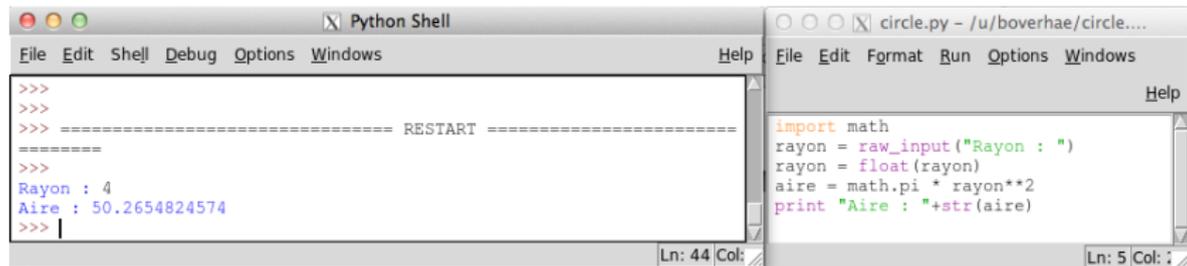
Au TP précédent, nous avons utilisé le **mode interactif** de Python, pratique pour tester de petites choses.

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

```
import math
rayon = raw_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)
- Ecrire 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



The screenshot shows two windows from the IDLE Python IDE. The left window, titled 'Python Shell', displays the interactive prompt with the following text:

```
>>>
>>>
>>> ===== RESTART =====
>>>
Rayon : 4
Aire : 50.2654824574
>>> |
```

The right window, titled 'circle.py - /u/boverhae/circle...', shows the source code for a Python script:

```
import math
rayon = raw_input("Rayon : ")
rayon = float(rayon)
aire = math.pi * rayon**2
print "Aire : "+str(aire)
```

Both windows have menu bars with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'.

Démonstration

Fonctions

Une **fonction** est une séquence d'instructions qui a un nom. Elle peut **recevoir** en entrée des **arguments** et peut **renvoyer** une **valeur de retour**.

```
| longueur = len('SPAM')
```

```
'SPAM' → len → 4
```

On peut voir une fonction comme une **boite noire** qui effectue un travail.

- Ses arguments sont les **informations** dont elle a besoin pour faire son travail.
- Sa valeur de retour est le **résultat** de son travail.

Composition : un argument d'une fonction peut être toute expression compatible :

```
| x = math.sin(degrees / 360.0 * 2 * math.pi)  
| x = math.exp(math.log(x+1))
```

Trouver de la documentation

Le site `docs.python.org` contient la documentation de tous les modules officiels.

La fonction `help()` donne l'aide d'une fonction ou d'un module :

```
>>> import math
>>> help(math.sqrt) #affiche l'aide de math.sqrt()
                         #(q pour quitter)
>>> help(math)      #affiche l'aide de tout le module math
```

La fonction `dir()` affiche le contenu d'un objet :

```
>>> dir()
['__builtins__', '__doc__', '__name__', '__package__']
>>> a = 3
>>> dir()
['__builtins__', '__doc__', '__name__', '__package__', 'a']
>>> import math
>>> dir(math)
['__doc__', '__file__', '__name__', '__package__', 'acos',
'acosh', 'asin', 'asinh', 'atan', 'atan2', 'atanh',
'ceil', # etc.
```

Exercices

1 à 9.