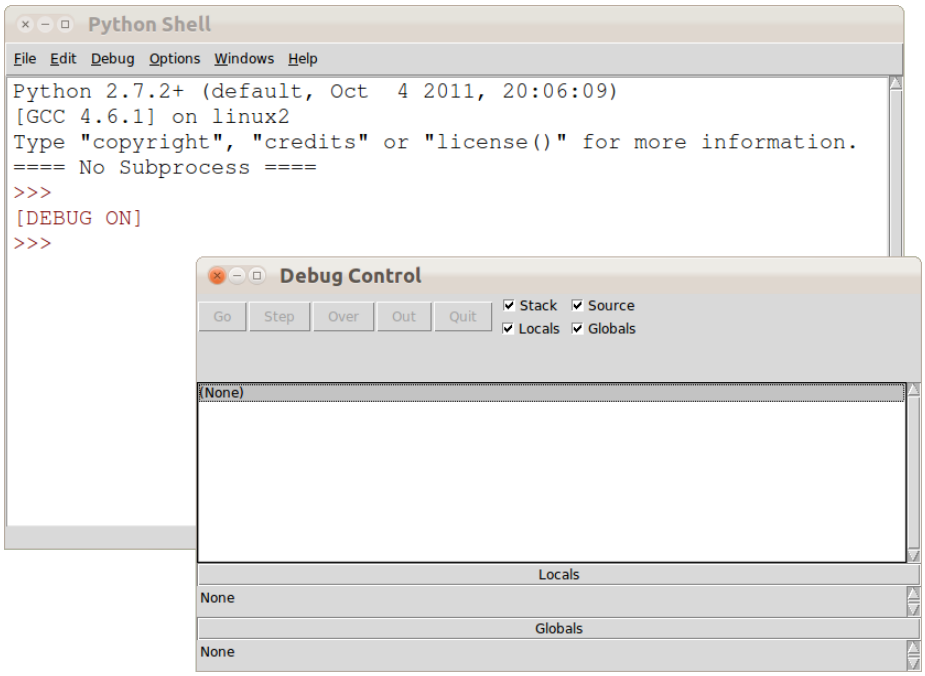
Le débogueur est un outil utile au débutant car il aide à comprendre le fonctionnement d'un script existant. Pour un professionnel, il permet le test et la mise au point d'un script en cours de développement (détection et élimination des bugs). Il existe plusieurs outils de déboguage sous Python, notamment le module standard `pdb` (en ligne de commande). Nous ne nous intéresserons qu'au débogueur de l'environnement IDLE (en mode graphique).

Exemple d'utilisation du débogueur

Commencer par télécharger le script test_debugger2.py Ce script affiche le carré des nombres entiers de 1 à 5. Nous allons tester son bon fonctionnement avec le débogueur de l'environnement IDLE. Ouvrir IDLE : Démarrer → Programmes → Python → IDLE (Python GUI) Puis lancer le débogueur : Debug → Debugger Cocher les cases Source et Globals :



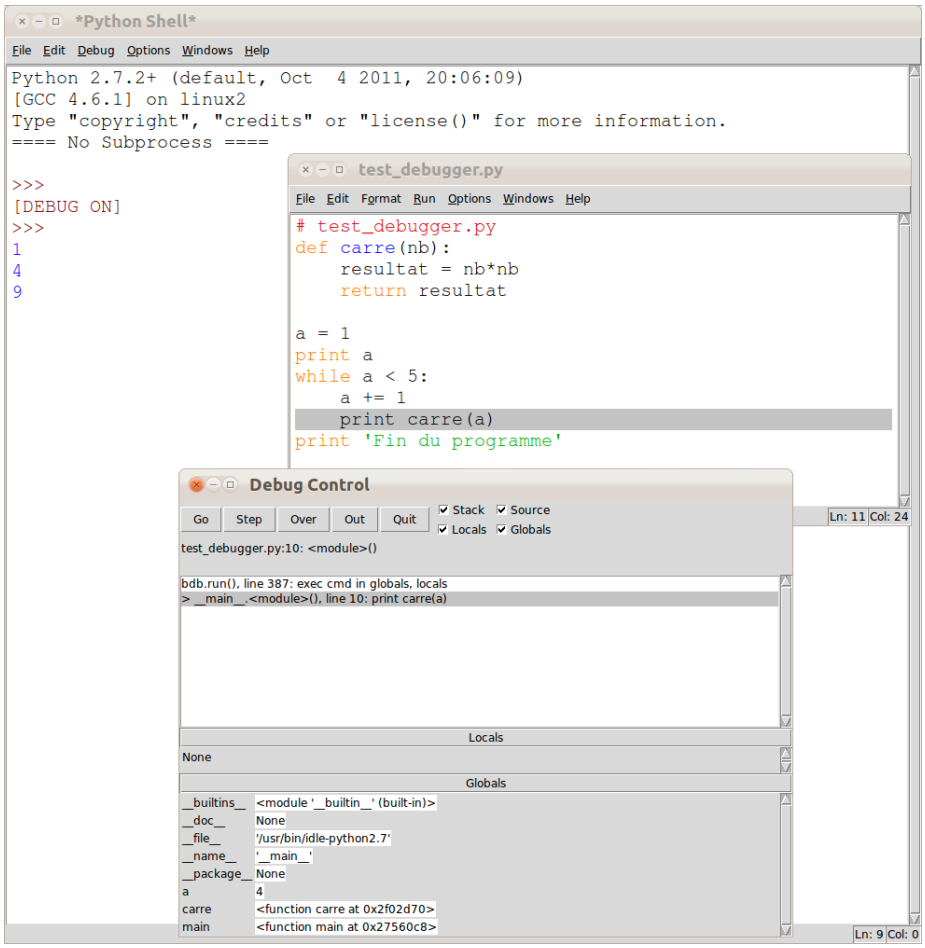
Le débogueur possède 5 boutons de commande :

- Go : Exécution normale du programme jusqu'au prochain point d'arrêt.
- Step : Exécution pas-à-pas (instruction par instruction)
- Over : Exécution pas-à-pas du programme principal (le débogueur ne rentre pas dans les fonctions)
- Out : Pour sortir de la fonction en cours
- Quit : Termine le programme

Dans l'interpréteur intéractif (Python Shell), ouvrir le script `test_debugger2.py` : File → Open → test_debugger2.py La fenêtre du code source s'ouvre. Dans

cette fenêtre : Run → Run Module (ou touche F5)

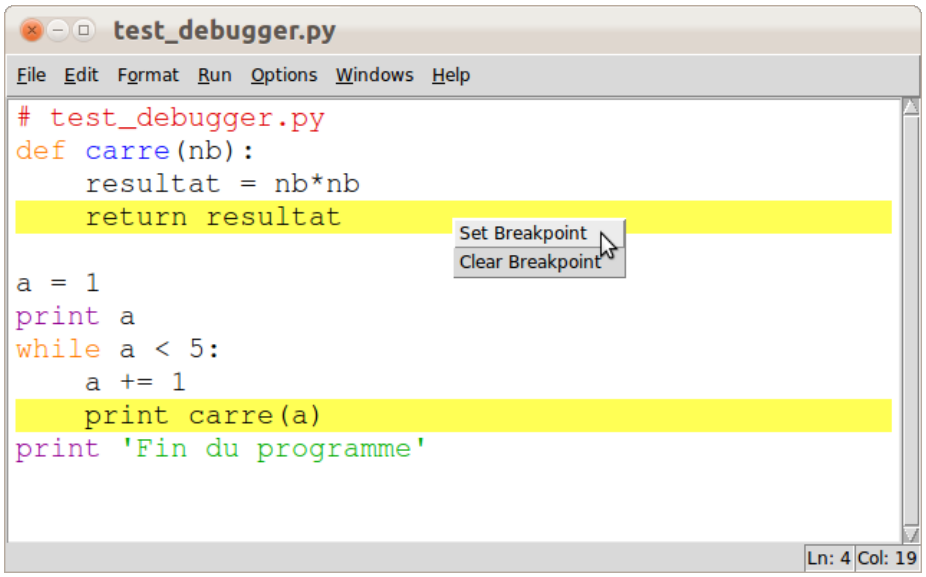
Pas-à-pas grossier Pour faire du pas-à-pas grossier, cliquer sur le bouton Over du débogueur :



Noter que l'on peut observer le contenu des variables (actuellement `a` vaut 4).

Pas-à-pas détaillé Pour faire du pas-à-pas détaillé, cliquer sur le bouton Step du débogueur. Pour sortir immédiatement d'une fonction, utiliser le bouton Out (en particulier pour sortir du script `PyShell.py` qui gère la fonction `print()`).

Point d'arrêt (Breakpoint) Dans la fenêtre du code source, sur la ligne d'instruction considérée, faire un clic droit et choisir Set Breakpoint (la ligne est alors surlignée en jaune) :



Puis utiliser le bouton Go.

Exercices

Exercice 11.1 A l'aide du débogueur, étudier la fonction récursive `factorielle()`qui retourne la factorielle d'un nombre entier :

```
def factorielle(x):
    if x < 2:
        return 1
    else:
        result = x*factorielle(x-1)
        return result

print(factorielle(5))
```

N.B. Une fonction récursive est une fonction qui s'appelle elle-même !

Exercice 11.2 A l'aide du débogueur, étudier la suite de Conway dont le script est disponible ici : conway2.py

```
>>>
0 1
1 11
2 21
3 1211
4 111221
5 312211
6 13112221
```

```
7 1113213211
8 31131211131221
9 13211311123113112211
10 11131221133112132113212221
...
```

Exercice 11.3 A l'aide du débogueur, étudier le module CompteBancaire du chapitre 5.

Webographie

- http://inventwithpython.com/chapter7.html
- http://www.dreamincode.net/forums/topic/210537-python-debugging-part-1
- Python documentation : Debugging and Profiling