UTILISATION DU DEBUGGER DE IDLE

Idle intègre un debugger parfois bien pratique lorsque l'on ne comprend ce qui ne va pas dans ce qui a été écrit.

Lancement du debugger

1. Pour l'utiliser, ouvrir le shell, puis aller dans les menus Debug > Debugger. Le shell renvoie alors une invite

[DEBUG ON]

et ouvre la fenêtre suivante :

| | Debug Control | ^ _ □ X | |
|--------|---------------------------------|---------|--|
| Go | Step Over Out Quit Stack Source | | |
| (None) | | | |
| Locals | | | |
| None | | | |

2. Cliquer sur Source afin de suivre lors d'une exécution pas à pas, le déroulement directement dans votre fichier.

3. Puis lancer l'interpréteur sur votre fichier ou une commande directement dans le shell.

Le Debugger vous propose alors une exécution pas à pas.

Le contenu des variables locales s'affichent au fur et à mesure dans le bas de la fenêtre.

Rôle des boutons

- Go : permet de lancer l'exécution du programme. Celle-ci s'arrêtera :
 - * lorsqu'elle rencontre un **point d'arrêt**, ajouté dans le script grâce au bouton droit de la souris puis Set Breakpoint, la ligne se surlignant alors en jaune,
 - * à la fin du programme,
 - * lorsque le programme demande une action de l'utilisateur.
- Step : permet d'exécuter la ligne courante du programme,
- Over : comme Step sauf que les fonctions utilisées dans la ligne courante ne sont pas exécutées pas à pas.

- Out : continue l'exécution jusqu'à la sortie de la fonction courante.
- Quit : stoppe l'exécution.

| Debug Control * - • × | sorts.py - /mnt/windows/Documents and Settings/jeremy/Dropb 🛧 💷 🗙 |
|---|--|
| Go Step Over Out Quit Stack Source | <u>File Edit Format Run Options Windows H</u> elp |
| Go Step Over Out Quit V Locals Globals sorts.py:21: selectionSort() "bdb'.run(), line 405: exec(cmd, globals, locals) "main_". <module>(), line 1: selectionSort(L) " '_main_':<module>(), line 1: selectionSort(L) " "</module></module> | <pre># Sorts /// L(i): L(i),L(j)=L(j),L(i) # Selection def indiceMin(L,i): n=len(L) mini=L(i) indice=i for k in range(i+1,n):</pre> |
| Locais | <pre>if mini>L[k]: mini=L[k] indice=k return indice def selectionSort(L): n=len(L) for i in range(n): echange(L,i,indiceMin(L,i_Cut Copy # somme(n-i-1 pour i entre 0 et leater k pour</pre> |
| <pre>Python 3.3.3 Shell</pre> | <pre># = n(n-1)/2 comparaisons. # Bubble def parcours(L): n=len(L) changement=False for i in range(n-1): if L[i)>L[i+1]: echange(L,i,i+1) changement=True return changement def bubbleSort(L): fini=False while not fini: fini=parcours(L) # Au pire : triée dans l'ordre inverse : n fois n-1 # Au mieux : déjà triée 1 fois n-1 comparaisons = n # Insertion def insert(L,i): indice=i while indice>0 and L[indice-1]>L[indice]: echange(L,indice, indice-1) indice=1</pre> |
| In: 9 Col: 0 | I n: 20 Col: 22 |