Python Standard Library Debugger (pdb)

Start the debugger by inserting a **`breakpoint()`** in your code. For example:

```
def bubblesort(lst):
    '''Bubblesort, more sparse and easier to debug'''
    breakpoint()
    num_el = len(lst)
    num_passes = num_el - 1
```

Common debugger commands:

command	description
p	Print the value of an expression.
pp	Pretty-print the value of an expression.
n	Continue execution until the next line in the current function is reached or it returns.
s	Step into a function/class
c	Continue execution and only stop when a breakpoint is encountered.
unt	Continue execution until the line with a number greater than the current one is reached. With a line number argument, continue execution until a line with a number greater or equal to that is reached.
l	List source code for the current file. Without arguments, list 11 lines around the current line or continue the previous listing.
ll	List the whole source code for the current function or frame.
b	With no arguments, list all breaks. With a line number argument, set a breakpoint at this line in the current file
w	Print a stack trace, with the most recent frame at the bottom. An arrow indicates the current frame.
u	Move the current frame count (default one) levels up in the stack trace (to an older frame).
d	Move the current frame count (default one) levels down in the stack trace (to a newer frame).
h	See a list of available commands.
h pdb	Show the full pdb documentation.
q	Quit the debugger and exit.
!	Will override a command to see a variable value. For example, to see the value of variable n instead of n for next: !n