经典使用:

在需要调试的地方加入代码:

```
import pdb ; pdb.set_trace()
```

使用参数:

a. h（elp）[命令]

没有参数:打印可用命令的列表。

有参数:打印关于该命令的帮助。例如:h n

b. b（reak）[[filename:] lineno | 函数 [, 条件]]

lineno:在当前文件中lineon行设置一个中断。

没有参数:列出所有中断，包括每个断点，断点被击中的次数，当前的忽略计数，以及相关的条件（如果有的话）。

c. tbreak [[filename:] lineno | 函数 [, 条件]]

临时断点，当它被首次击中时被自动删除。参数与break一样。

d. cl（ear）[filename: lineno | bpnumber [bpnumber …]]

文件名: lineno : 清除此行中的所有断点。

无参:清除所有断点

e. s(tep)

执行当前行，有函数就进入函数。

f. n(ext)

继续执行，直到当前功能中的下一行达到或返回。

g. r(eturn)

继续执行，直到当前函数返回。

h. c(ont(inue))

继续执行，仅在遇到断点时停止。

i. j（ump）lineno

跳到指定行

j. l（ist）[first [, last]]

列出当前文件的源代码。默认显示11行

k. a（rgs）

打印当前函数的参数列表。

l. p 变量名(表达式)

打印变量名的值

print也可以使用，但不是调试器命令 - 这将执行Python print语句。

m. pp 表达式

像p命令一样，除了该表达式的值使用该pprint模块漂亮打印。

n. q(uit)/exit

退出调试,quit比较暴力