python 退出程序的方式

python 程序退出方式 [sys.exit() os._exit() os.kill() os.popen(...)]

1. `sys.exit()`
   执行该语句会直接退出程序，这也是经常使用的方法，也不需要考虑平台等因素的影响，一般是退出Python程序的首选方法。
   该方法中包含一个参数status，默认为0，表示正常退出，也可以为1，表示异常退出：

```
import sys
sys.exit()
sys.exit(0)
sys.exit(1)
```

   该方法引发的是一个SystemExit异常(这是唯一一个不会被认为是错误的异常)，当没有设置捕获这个异常将会直接退出程序执行，当然也可以捕获这个异常进行一些其他操作。

   个人实际在 win 平台使用，退出程序都会抛出异常。

2. `os._exit()`
   效果也是直接退出，不会抛出异常，但是其使用会受到平台的限制，但我们常用的Win32平台和基于UNIX的平台不会有所影响。
   有说是调用了 C 语言的 `_exit()` 函数

   最终使用了这个，无异常，结束进程。

3. `os.kill()`
   一般用于直接Kill掉进程，但是只能在UNIX平台上有效。
   基本原理：该函数是模拟传统的 UNIX 函数发信号给进程，其中包含两个参数：一个是进程名，即所要接收信号的进程；一个是所要进行的操作。
   操作（第二个参数）的常用取值为：

```
SIGINT      终止进程     中断进程
SIGTERM     终止进程     软件终止信号
SIGKILL     终止进程     杀死进程
SIGALRM     闹钟信号
```

   有平台限制。

4. Windows 下 Kill 进程
   这里使用的是`os.popen()`, 该方法是用于直接执行系统命令，而在 Windows 下其实就是使用 taskkill 来 kill 掉进程，其基本形式是：

```
taskkill    /pid    程序的PID号码
```

可以直接在 CMD 窗口下试下这个命令.... 可以先打开一个计算器程序，然后使用 tasklist 查看该程序的 pid, 这里是711.

```
import os
if __name__ == "__main__":
    pid = 711
    os.popen('taskkill.exe /pid:'+str(pid))
```

Reference

python 退出程序的方式

2018.12.12