　调用模块的实质是运行python代码，比如a.py文件里有函数f()，那么你在调用a模块的时候，实质是运行了a模块里的函数f()，这个时候内存里就有这个函数了，可以直接用，那是不是随便一个py类型的文件都可以直接调用呢？不是的，调用模块的优先级：先从当前目录里面找你需要调用的模块，当前目录找不到的话就python自己的环境变量里找。

　　下面例出常用的os模块的方法

import os,sys

print(sys.path)#获取python的环境变量,以list形式返回

#输出：['E:\\study\\Automantic\\jxz-code\\Course4']

print(os.listdir('./'))#获取指定目录下的文件及文件夹名称，以list形式返回

#输出：['access.log', 'b.txt', 'c.txt', 'course4作业.py', 'goods.txt', 'user\_info.txt', '、', '函数.py']

print(os.getcwd())#获取当前目录

#输出：E:\study\Automantic\jxz-code\Course4

# print(os.chdir('E:\study\Automantic\jxz-code'))#更换当前目录

print(os.rename('c.txt','a.txt'))#修改文件名称

print(os.mkdir('新目录'))#创建文件夹

print(os.rmdir('新目录'))#删除文件夹（只能删除空文件夹）

print(os.makedirs('E:\\xixi\\haha'))#依次创建目录

print(os.removedirs('E:\\xixi\\haha'))#依次删除非空目录

print(os.sep)#获取当前操作系统的路径分隔符

#输出：\

print(os.environ)#获取当前操作系统的环境变量

#输出：environ({'ALLUSERSPROFILE': 'C:\\ProgramData'})

print(os.pathsep)#获取当前系统的环境变量中每个路径的分隔符，linux是:，windows是;

#输出：;

print(os.path.abspath(\_\_file\_\_))#获取当前文件的绝对路径

#输出：E:\study\Automantic\jxz-code\Course4\函数.py

print(os.path.dirname(os.path.abspath(\_\_file\_\_)))#获取指定路径的父目录

#输出：E:\study\Automantic\jxz-code\Course4

print(os.path.isdir(os.path.abspath(\_\_file\_\_)))#判断指定路径是不是一个文件夹

#输出：False

print(os.path.isfile(os.path.abspath(\_\_file\_\_)))#判断指定路径是不是一个文件

#输出：True

print(os.path.join('一级','二级','三级','haha.txt'))#将内容以当前操作系统的路径分隔符拼接成一个路径

#输出：一级\二级\三级\haha.txt

print(os.path.split('E:\study\Automantic\jxz-code\Course4\函数.py'))#分割路径和文件名

#输出：('E:\\study\\Automantic\\jxz-code\\Course4', '函数.py')

print(os.path.exists('E:\study\Automantic\jxz-code\Course4\函数.py'))#判断目录或文件是否存在

#输出：True

　　os和sys的应用:

　　问题：A君写了一个小系统，模块之间彼此可以正常调用，但是B君拷走了A君的代码，运行各种报错，大多为引用模块不存在，函数、方法未定义等，这是因为A君运行的时候都已经把他的绝对路径加进了python的环境变量，而B君在运行的时候并没有加环境变量，运行程序，在调用模块时，找不到模块自然报错了。

　　解决：上面的模块介绍里可以看出，sys.path可以获取到一个python环境变量的列表，那么我们只需要把代码的工作目录加到列表里就可以了，具体代码如下：

import sys,os

cwd=os.path.abspath(\_\_file\_\_) #当前文件的绝对路径

pwd=os.path.dirname(cwd)#获取当前文件的父级目录

ppwd=os.path.dirname(pwd)#获取当前文件的父级目录的父级目录

sys.path.insert(0,ppwd)#将当前文件的父级目录的父级目录加入python环境变量，那么这个文件夹下的所有.py文件都可以当做模块引入了

from conf.setting import LOGS\_PATH,FILE\_PATH

from lib.wrong import write

write(LOGS\_PATH,FILE\_PATH)

　　上题的截图：

