不同的script执行方式会造成不一样的结果，尤其对bash的环境影响很大，脚本的执行方式有大概以下几种方式。

首先写一个简单的test.sh文件：

#!/bin/bash

read -p "Please input your first name:" firstname

read -p "Please input your last name:" lastname

echo -e "\nYour full name is: $firstname $lastname"

**使用sh test.sh执行**

**使用 sh test.sh 来执行script文件，该方法标明使用 sh 这种shell来执行test.sh文件**，sh已经是一种被bash替代的shell，尽管我们在test.sh中声明使用 #!/bin/bash 来执行我们的文件，但此时使用sh而不是bash，则#!/bin/bash 已不起作用。

**使用bash test.sh 执行**

该方法其实与 sh test.sh 的原理一样，只是使用了 /bin/bash 该种shell来执行我们的脚本文件。

所以，其实使用 dash test.sh' 也是可以的，只是取决于自己想使用那种shell来执行脚本，但sh、bash、dash三者有些许差别，对于部分关键字如 let，bash支持，而sh和dash并不支持，对于部分关键字则选择使用bash。

**使用点 . 执行**

该种方式使用之前必须为文件添加执行的权限：

$ chmod +x test.sh

添加完执行权限之后，便可以使用 ./test.sh 来执行脚本，该方式与 bash test.sh 是一样的 ，默认使用 bin/bash 来执行我们的脚本。

只有该种执行方式需要对文件添加执行权限，其他方式并不需要。

**使用source执行**

使用source则也能够直接执行我们的脚本：

source test.sh

**区别**

当我们使用 sh test.sh 、bash test.sh 、 ./test.sh 执行脚本的时候，该test.sh运行脚本都会使用一个新的shell环境来执行脚本内的命令，也就是说，**使用这3种方式时，其实script是在子进程的shell内执行，当子进程完成后，子进程内的各项变量和操作将会结束而不会传回到父进程中**。

无法理解？？看下面例子：

[root@ubuntu] *# bash test.sh*

Please input your first name: yao <==输入firstname

Please input your last name: pentonBin <==输入lastname

Your full name is: yao pentonBin

[root@ubuntu] *# echo $firstname*

<==这里没有输出

如果使用source方法来执行脚本呢？

[root@ubuntu] *# source test.sh*

Please input your first name: yao <==输入firstname

Please input your last name: pentonBin <==输入lastname

Your full name is: yao pentonBin

[root@ubuntu] *# echo $firstname*

yao <==这里输出firstname

也就是说，**source方法执行脚本是在父进程中执行的**，test.sh的各项操作都会在原本的shell内生效，这也是为什么不注销系统而要让某些写入～/.bashrc的设置生效时，需要使用 source ~/.bashrc 而不能使用 bash ~/.bashrc