[Top](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/ENGINEER/DAY01/CASE/01/index.html" \l "page_top_case)

# NSD ENGINEER DAY01

1. [案例1：Shell脚本的编写及测试](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/ENGINEER/DAY01/CASE/01/index.html" \l "case1)
2. [案例2：重定向输出的应用](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/ENGINEER/DAY01/CASE/01/index.html" \l "case2)
3. [案例3：使用特殊变量](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/ENGINEER/DAY01/CASE/01/index.html" \l "case3)
4. [案例4：编写一个判断脚本](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/ENGINEER/DAY01/CASE/01/index.html" \l "case4)
5. [案例5：编写一个批量添加用户脚本](http://tts.tmooc.cn/ttsPage/LINUX/NSDTN201904/ENGINEER/DAY01/CASE/01/index.html" \l "case5)

## 1 案例1：Shell脚本的编写及测试

### 1.1 问题

本例要求两个简单的Shell脚本程序，任务目标如下：

1. 编写一个面世问候 /root/helloworld.sh 脚本，执行后显示出一段话“Hello World！！”
2. 编写一个能输出系统信息的 /root/sysinfo 脚本，执行后依次输出当前红帽系统的版本信息、当前使用的内核版本、当前系统的主机名

### 1.2 方案

规范Shell脚本的一般组成：

1. #! 环境声明（Sha-Bang）
2. # 注释文本
3. 可执行代码

### 1.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：编写helloworld.sh问候脚本

1）编写脚本代码

1. [root@server0 ~]# vim /root/helloworld.sh
2. #!/bin/bash
3. echo "Hello World !!"

2）添加x执行权限

1. [root@server0 ~]# chmod +x /root/helloworld.sh

3）运行脚本测试

1. [root@server0 ~]# /root/helloworld.sh
2. Hello World !!

步骤二：编写sysinfo系统信息报告脚本

1）编写脚本代码

1. [root@server0 ~]# vim /root/sysinfo
2. #!/bin/bash
3. cat /etc/redhat-release
4. uname -r
5. hostname

2）添加x执行权限

1. [root@server0 ~]# chmod +x /root/sysinfo

3）运行脚本测试

1. [root@server0 ~]# /root/sysinfo
2. Red Hat Enterprise Linux Server release 7.0 (Maipo)
3. 3.10.0-123.el7.x86\_64
4. server0.example.com

## 2 案例2：重定向输出的应用

### 2.1 问题

本例要求编写一个脚本 /root/out.sh，功能特性如下：

1. 执行此脚本显示 I love study !!
2. 执行 /root/out.sh 2> err.log 应该没有显示，但是查看 err.log 文件的内容为 I love study !!

### 2.2 方案

屏幕输出文本的类别：

* 标准输出（1）：命令行执行正常的显示结果
* 标准错误（2）：命令行执行出错或异常时的显示结果

将屏幕显示信息保存到文件：

* cmd > file 、 cmd >> file
* cmd 2> file 、 cmd 2>> file
* cmd &> file 、cmd 2> file 1>&2

使用1>&2或>&2操作，可以将命令行的标准输出编程标准错误。

### 2.3 步骤

实现此案例需要按照如下步骤进行。

步骤：编写out.sh输出测试脚本

1）编写脚本代码

1. [root@server0 ~]# vim /root/out.sh
2. #!/bin/bash
3. echo "I love study !!" >&2

2）添加x执行权限

1. [root@server0 ~]# chmod +x /root/out.sh

3）运行脚本测试

1. [root@server0 ~]# /root/out.sh
2. I love study !!
3. [root@server0 ~]# /root/out.sh 2> err.log
4. [root@server0 ~]# cat err.log
5. I love study !!

## 3 案例3：使用特殊变量

### 3.1 问题

本例要求编写一个脚本 /root/myuseradd，功能特性如下：

1）此脚本可接收2个位置参数，能够按照下列格式执行：

1. /root/myuseradd 用户名 密码

2）此脚本执行后，能显示“一共提供了 $# 个参数”，然后在下一行显示“用户名是 $1，密码是 $2 ”，紧跟下一行开始输出对应文件的前几行内容。

### 3.2 方案

使用位置变量可以取得在执行脚本时提供的命令行参数：

* 表示为 $n，n为序号
* $1、$2、.. .. ${10}、${11}、.. ..

使用预定义变量$#可以统计执行脚本时提供的位置变量个数。

### 3.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：编写 /root/myuseradd 添加用户的脚本

1）编写脚本代码

1. [root@server0 ~]# vim /root/myuseradd
2. #!/bin/bash
3. echo "一共提供了 $# 个参数"
4. echo "用户名是 $1，密码是 $2 "
5. useradd $1
6. echo "$2" | passwd --stdin $1

2）添加x执行权限

1. [root@server0 ~]# chmod +x /root/myuseradd.sh

步骤二：测试 /root/myuseradd 脚本

1）测试添加用户 bob，密码设为 1234567

1. [root@server0 ~]# /root/myuseradd bob 1234567
2. 一共提供了 2 个参数
3. 用户名是 bob，密码是 1234567
4. 更改用户 bob 的密码 。
5. passwd：所有的身份验证令牌已经成功更新。
6. [root@server0 ~]# id bob
7. uid=1002(bob) gid=1002(bob) 组=1002(bob)

2）测试添加用户 jerry，密码设为 1234567

1. [root@server0 ~]# /root/myuseradd jerry 1234567
2. 一共提供了 2 个参数
3. 用户名是 jerry，密码是 1234567
4. 更改用户 jerry 的密码 。
5. passwd：所有的身份验证令牌已经成功更新。
6. [root@server0 ~]# id jerry
7. uid=1003(jerry) gid=1003(jerry) 组=1003(jerry)

## 4 案例4：编写一个判断脚本

### 4.1 问题

本例要求在虚拟机 server0 上创建 /root/foo.sh 脚本，任务目标如下：

1. 当运行/root/foo.sh redhat，输出为fedora
2. 当运行/root/foo.sh fedora，输出为redhat
3. 当没有任何参数或者参数不是 redhat 或者 fedora时，其错误输出产生以下信息： /root/foo.sh redhat|fedora

### 4.2 方案

Shell脚本中执行条件测试的方式：

* 任何一条命令行
* test 测试表达式
* [ 测试表达式 ]

常用的test测试选项：

* 文件状态检测 -f、-d、-e、-r、-w、-x
* 整数值比较 -gt、-ge、-eq、-ne、-lt、-le
* 字符串比较 ==、!=
* 取反操作 !

多分支if选择结构：

1. if 条件测试操作1;then
2. 命令序列1....
3. elif 条件测试操作2;then
4. 命令序列2....
5. else
6. 命令序列3....
7. fi

### 4.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：编写foo.sh判断脚本

1）编写脚本代码

1. [root@server0 ~]# vim /root/foo.sh
2. #!/bin/bash
3. if [ $# -eq 0 ];then
4. echo "/root/foo.sh redhat|fedora" >&2
5. elif [ $1 = "redhat" ];then
6. echo "fedora"
7. elif [ $1 = "fedora" ];then
8. echo "redhat"
9. else
10. echo "/root/foo.sh redhat|fedora" >&2
11. fi

2）添加x执行权限

1. [root@server0 ~]# chmod +x /root/foo.sh

步骤二：测试foo.sh判断脚本

1）测试提供正确参数的情况

1. [root@server0 ~]# /root/foo.sh redhat
2. fedora
3. [root@server0 ~]# /root/foo.sh fedora
4. Redhat

2）测试提供非预期参数的情况

1. [root@server0 ~]# /root/foo.sh ubuntu
2. /root/foo.sh redhat|fedora

3）测试不提供参数的情况

1. [root@server0 ~]# /root/foo.sh
2. /root/foo.sh redhat|fedora

## 5 案例5：编写一个批量添加用户脚本

### 5.1 问题

本例要求在虚拟机 server0 上创建 /root/batchusers 脚本，任务目标如下：

1. 此脚本要求提供用户名列表文件作为参数
2. 如果没有提供参数，此脚本应该给出提示 Usage: /root/batchusers，退出并返回相应值
3. 如果提供一个不存在的文件，此脚本应该给出提示 Input file not found，退出并返回相应值
4. 新用户的登录Shell为 /bin/false，无需设置密码
5. 列表测试文件：http://classroom/pub/materials/userlist

### 5.2 方案

单分支if选择结构：

1. if 条件测试操作
2. then
3. 命令序列....
4. fi

脚本的退出状态：取决于退出前最后一条命令的 $? 值，或者“exit 整数值”指定。

列表式for循环结构：

1. for 变量名 in 值1 值2 值3 .. ..
2. do
3. 命令序列（$变量名）
4. done

使用命令替换来获取命令结果：$(命令行)

### 5.3 步骤

实现此案例需要按照如下步骤进行。

步骤一：编写batchusers批量添加用户脚本

1）编写脚本代码

1. [root@server0 ~]# vim /root/batchusers
2. #!/bin/bash
3. if [ $# -eq 0 ] ; then
4. echo "Usage: /root/batchusers <userfile>" >&2
5. exit 1
6. fi
7. if [ ! -f $1 ] ; then
8. echo "Input file not found" >&2
9. exit 2
10. fi
11. for name in $(cat $1)
12. do
13. useradd -s /bin/false $name
14. done

2）添加x执行权限

1. [root@server0 ~]# chmod +x /root/batchusers

步骤二：测试batchusers批量添加用户脚本

1）下载用户列表测试文件：

1. [root@server0 ~]# wget http://classroom/pub/materials/userlist -O /root/userlist
2. .. ..
3. 2016-11-27 17:23:32 (2.83 MB/s) - ‘/root/userlist’ saved [27/27]
4. [root@server0 ~]# cat /root/userlist                     //检查下载文件
5. duanwu
6. zhongqiu
7. zhsan
8. lisi

2）实现批量添加用户：

1. [root@server0 ~]# /root/batchusers /root/userlist
2. [root@server0 ~]# id duanwu
3. uid=1006(duanwu) gid=1006(duanwu) groups=1006(duanwu)

3）测试其他异常处理：

1. [root@server0 ~]# /root/batchusers                         //未提供列表文件
2. Usage: /root/batchusers <userfile>
3. [root@server0 ~]# echo $?
4. 1
5. [root@server0 ~]# /root/batchusers /root/userlist.txt        //提供的列表文件找不到
6. Input file not found
7. [root@server0 ~]# echo $?
8. 2