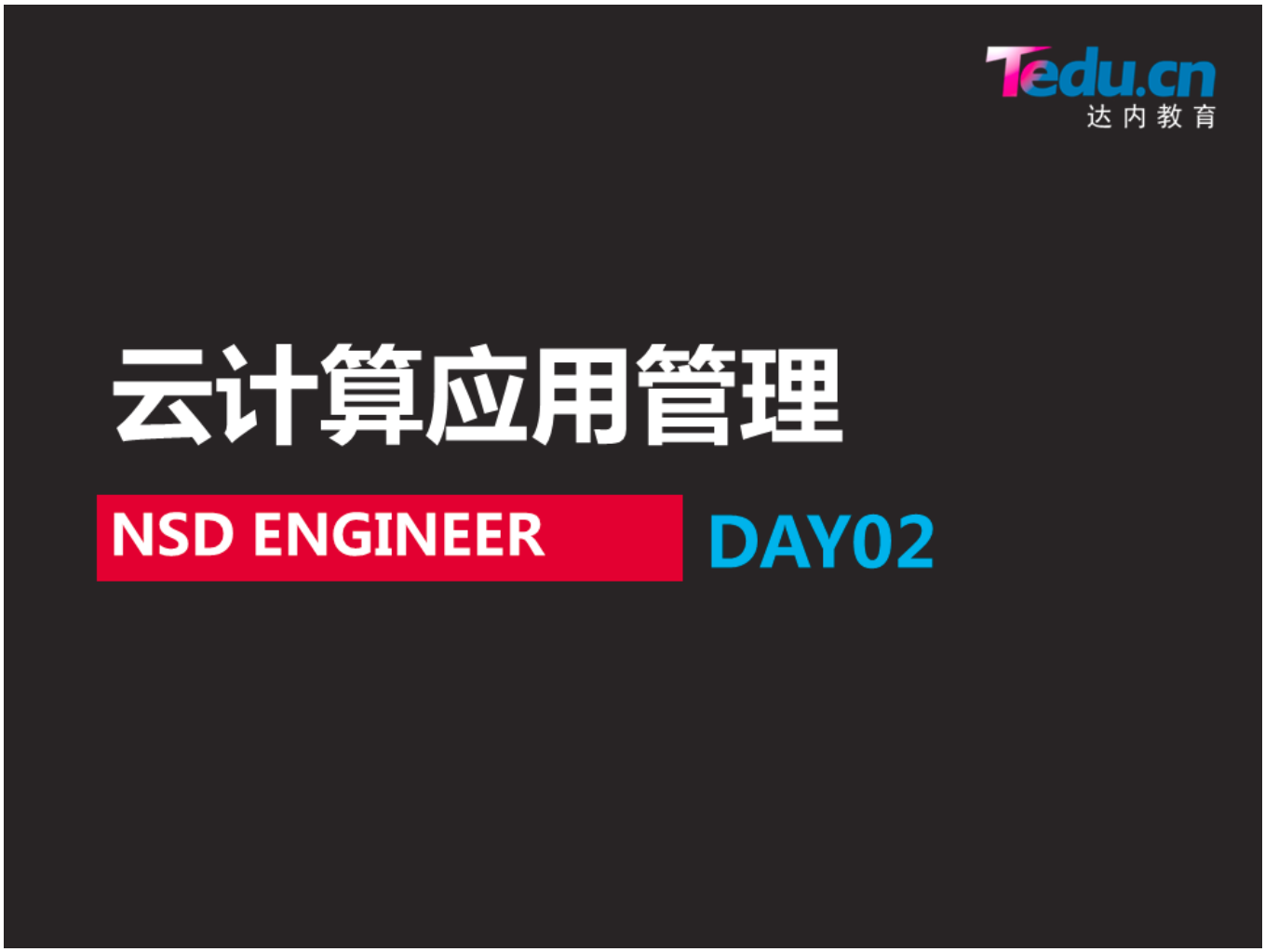
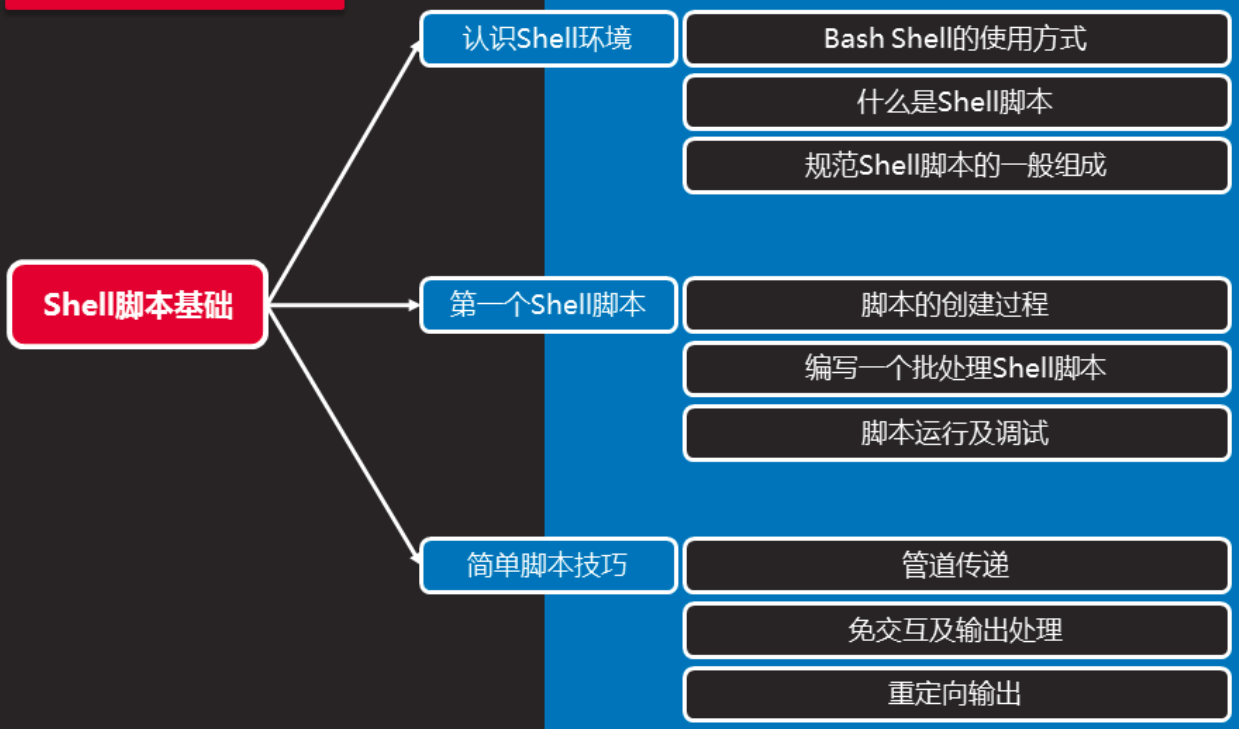
ENGINEER DAY02



内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	Shell脚本基础
	10:30 ~ 11:20	使用变量
	11:30 ~ 12:00	
下午	14:00 ~ 14:50	条件测试及选择
	15:00 ~ 15:50	
	16:10 ~ 17:00	列表式循环
	17:10 ~ 18:00	总结和答疑

Shell脚本基础



认识Shell环境

Bash Shell的使用方式

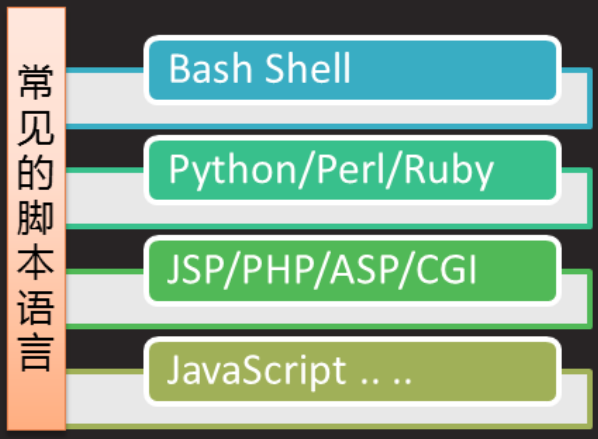
知识讲解

- 交互式
  - 人工干预、智能化程度高
  - 逐条解释执行、效率低
- 非交互式
  - 需要提前设计、智能化难度大
  - 批量执行、效率高
  - 方便在后台静悄悄地运行

什么是Shell脚本

知识讲解

- 提前设计可执行语句，用来完成特定任务的文件
  - 解释型程序
  - 顺序、批量执行



规范Shell脚本的一般组成

知识讲解

- #! 环境声明（Sha-Bang）
- # 注释文本
- 可执行代码

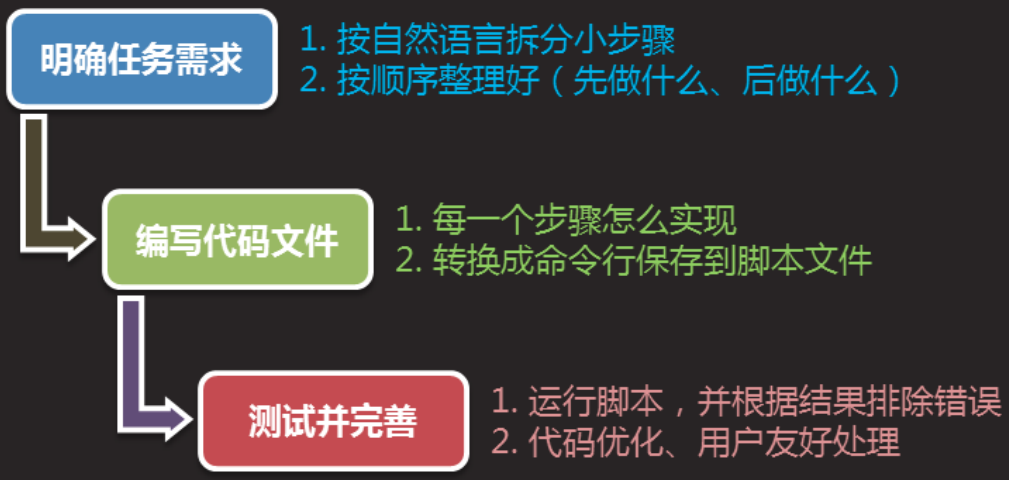
```
[root@server0 ~]# vim  /etc/rc.local
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
.. ..
touch /var/lock/subsys/local
[ -f /etc/rc.d/rc.hostname_hack ] && source /etc/rc.d/rc.hostna
me_hack
.. ..
```

第一个Shell脚本

脚本的创建过程

- 如何写出自己的第一个Shell脚本？

知识讲解



编写一个批处理Shell脚本

Tedu.cn 达内教育

知识讲解

- 任务需求：依次输出以下系统信息
  - 红帽系统版本、内核版本、当前的主机名
- 转换成代码
  - cat /etc/redhat-release、uname -r、hostname

```
[root@server0 ~]# vim /root/sysinfo
#!/bin/bash
cat /etc/redhat-release
uname -r
hostname
```

脚本运行及调试

Tedu.cn 达内教育

知识讲解

- 添加执行权限，独立运行

```
[root@server0 ~]# chmod +x /root/sysinfo
[root@server0 ~]# /root/sysinfo
Red Hat Enterprise Linux Server release 7.0 (Maipo)
3.10.0-123.el7.x86_64
server0.example.com
```

- 通过 sh 执行脚本代码
  - 结合 -x 选项，可以调试脚本代码

```
[root@server0 ~]# sh -x /root/sysinfo
+ cat /etc/redhat-release                          //代码
Red Hat Enterprise Linux Server release 7.0 (Maipo)  //输出结果
+ uname -r
.. ..
```

案例1：Shell脚本的编写及测试

课堂练习

1. 编写一个面世问候 /root/helloworld.sh 脚本
   - 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
   1）输出当前红帽系统的版本信息
   2）输出当前使用的内核版本
   3）输出当前系统的主机名

简单脚本技巧

管道传递

知识讲解

- 使用 | 管道操作
  - 将前一条命令的标准输出交给后一条命令处理
  - cmd1 | cmd2 [| cmd3] .. ..

```
[root@server0 ~]# ls --help | less          //传递多屏文本
.. ..
[root@server0 ~]# echo 'Mail Data' | mail -s Test1 student
.. ..                                       //传递邮件正文内容
```

免交互及输出处理

知识讲解

- 免交互处理
  - 脚本一般在后台执行，要尽量减少人工交互的语句
- 脚本输出信息的处理
  - 记录有价值的信息（>> /var/log/foo.log）
  - 屏蔽无价值的、干扰性的信息（&> /dev/null）
  - 自定义输出：echo '文本字符串'

重定向输出

知识讲解

- 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

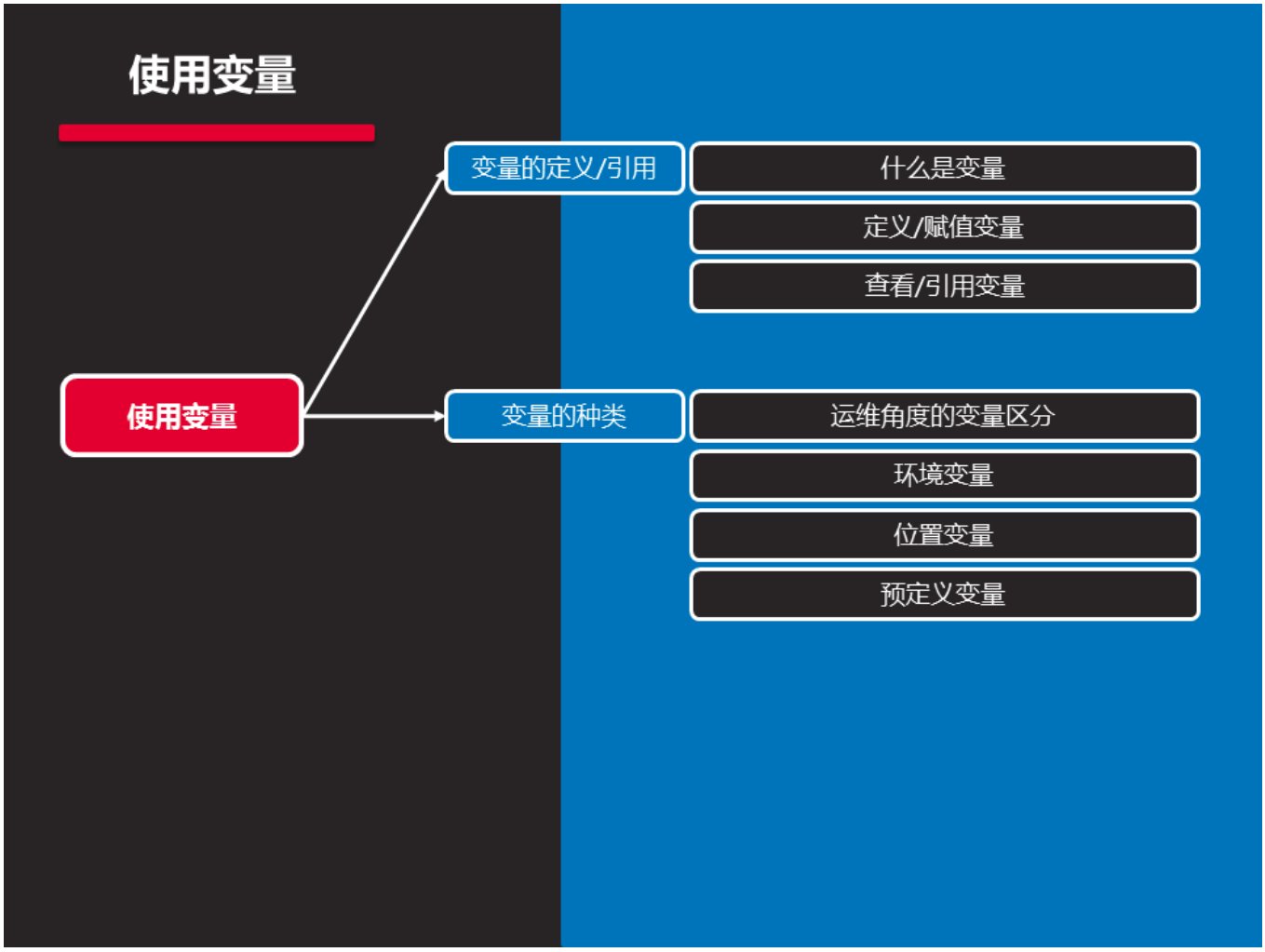
案例2：重定向输出的应用

课堂练习

编写一个脚本 /root/out.sh ，要求如下：

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 !!


使用变量
使用变量
变量的定义/引用
什么是变量
定义/赋值变量
查看/引用变量
变量的种类
运维角度的变量区分
环境变量
位置变量
预定义变量


Tedu.cn
达内教育
变量的定义/引用

什么是变量

知识讲解

- 以不变的名称存放的可能会变化的值
  - 变量名=变量值
  - 方便以固定名称重复使用某个值
  - 提高对任务需求、运行环境变化的适应能力

```
[student@server0 ~]$ echo  $USER
student                                  //环境1的取值

[root@server0 ~]# echo  $USER
root                                     //环境2的取值
```

定义/赋值变量

知识讲解

- 设置变量时的注意事项
  - 若指定的变量名已存在，相当于为此变量重新赋值
  - 等号两边不要有空格
  - 变量名由字母/数字/下划线组成，区分大小写
  - 变量名不能以数字开头，不要使用关键字和特殊字符

```
[root@server0 ~]# X=12                   //变量 X 的值 12
[root@server0 ~]# var1=CentOS            //变量 var1 的为 CentOS
```

查看/引用变量

Tedu.cn 达内教育

知识讲解

- 基本格式
  - 引用变量值：$变量名
  - 查看变量值：echo $变量名、echo ${变量名}

```
[root@svr5 ~]# echo $var16.5                    //未定义的变量为空值
.5
[root@svr5 ~]# echo ${var1}6.5                  //以 {} 界定易混淆名称
CentOS6.5
```

Tedu.cn 达内教育

变量的种类

运维角度的变量区分

Tedu.cn 达内教育

知识讲解

- 根据变量的用途不同区分

类 型	说 明
环境变量	变量名一般都大写，用来设置用户/系统环境
位置变量	bash内置，存储执行脚本时提供的命令行参数
预定义变量	bash内置，可直接调用的特殊值，不能直接修改
自定义变量	用户自主设置、修改及使用

环境变量

Tedu.cn 达内教育

知识讲解

- 常见的环境变量
  - PWD、PATH、USER 、LOGNAME
  - SHELL、HOME

```
[root@server0 ~]# echo  $PATH
/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin

[root@svr7 ~]# which  nmcli                    //查找命令程序存在哪
/usr/bin/nmcli
```

位置变量

知识讲解

- 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}、.. ..

```
[root@server0 ~]# cat  /root/mycat
#!/bin/bash
cat  $1
[root@server0 ~]# /root/mycat  /etc/redhat-release
Red Hat Enterprise Linux Server release 7.0 (Maipo)
```

预定义变量

知识讲解

- 用来保存脚本程序的执行信息
  - 直接使用这些变量
  - 不能直接为这些变量赋值

变量名	含 义
$#	已加载的位置变量的个数
$*	所有位置变量的值
$?	程序退出后的状态值，0表示正常，其他值异常

案例3：使用特殊变量

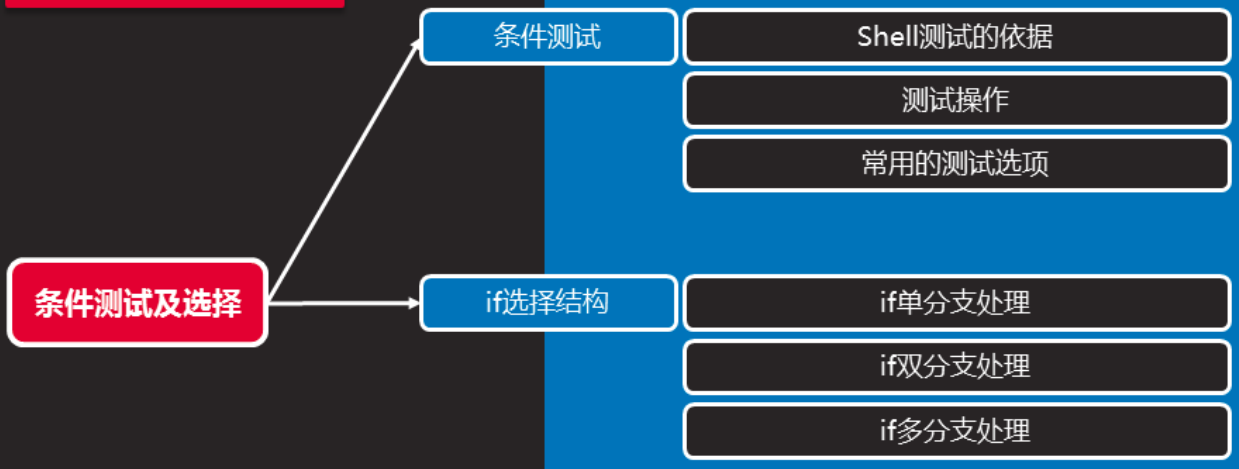
Tedu.cn 达内教育

课堂练习

编写一个脚本 /root/myuseradd，要求如下：

- 接收2个位置参数，添加指定用户并设置密码：
  /root/myuseradd 用户名 密码
- 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 ，然后成功添加此用户账号

条件测试及选择



条件测试

Shell测试的依据

知识讲解

- 命令行/程序的退出状态值 $?
  - 值为 0 ，表示执行成功
  - 值不为 0 ，表示执行异常或失败
- 在脚本中自定义退出状态值
  - 以退出之前最后一条命令的 $? 作为脚本的退出状态值
  - 也可以自行替换，添加 exit 整数

测试操作

知识讲解

- 几种可用的测试方式
  - 正常的命令行
  - test -选项 参数...
  - [测试表达式]

```
[root@server0 ~]# id fleyd &> /dev/null          //测试操作
[root@server0 ~]# echo $?
1                                                //状态值为1，表示失败

[root@server0 ~]# [ 100 -gt 50 ]                 //测试表达式
[root@server0 ~]# echo $?
0                                                //状态值为0，表示成功
```

常用的测试选项

知识讲解

- 检查文件状态
  - -e、-d、-f、-r、-w、-x
- 比较整数大小
  - -gt、-ge、-eq、-ne、-lt、-le
- 字符串比对
  - ==、!=

```
[root@server0 ~]# [ $USER == 'student' ]
[root@server0 ~]# echo $?
1                                    //说明当前用户不是 student
```

if选择结构

if单分支处理

- 单分支：当条件满足时，作xx处理
  - 如果服务器CPU负载超标，发送告警邮件

知识讲解

```
if 条件测试 ; then
    命令序列xx
fi
```

if双分支处理

Tedu.cn 达内教育

知识讲解

- 双分支：当条件满足/不满足时，分别作xx、yy处理
  - 如果软件包foo未装，则将其安装
  - 否则，放弃安装

```
if 条件测试 ; then
   命令序列xx
else
   命令序列yy
fi
```

if多分支处理

Tedu.cn 达内教育

知识讲解

- 多分支：当条件1满足时，作xx处理；否则继续检查条件2，若成立则作yy处理；否则，作zz处理
  - 如果软件包foo未装，则将其安装
  - 否则，如果已装的软件包foo版本比较旧，则将其升级
  - 再否则，放弃安装

```
if 条件测试1 ; then
   命令序列xx
elif 条件测试2 ; then
   命令序列yy
else
   命令序列zz
fi
```

案例4：编写一个判断脚本

Tedu.cn 达内教育

课堂练习

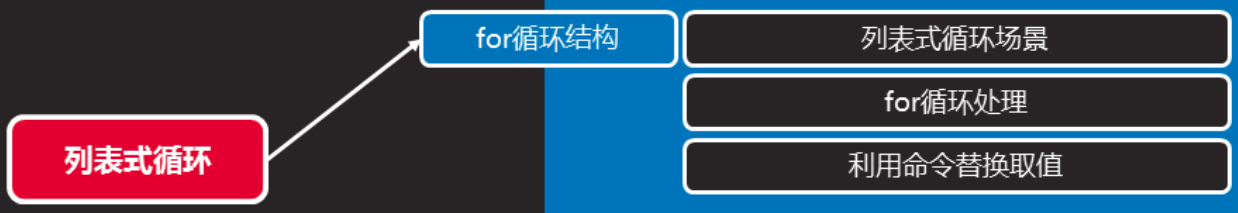
在 server0 上创建 /root/foo.sh 脚本

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

列表式循环



for循环结构

列表式循环场景

知识讲解

- 给定一批对象，反复执行类似的操作
  - 献血人 ==> 采血
  - 购票者 ==> 售票

for循环处理

Tedu.cn 达内教育

知识讲解

- 遍历/列表式循环
  - 根据变量的不同取值，重复执行xx处理

```
for  变量名  in  值列表
do
   命令序列
done
```

利用命令替换取值

Tedu.cn 达内教育

知识讲解

- 使用 $(命令行) 操作
  - 先执行括号内的命令行，提取此命令行的标准输出
  - 然后将标准输出结果替换整个 $() 表达式
  - 可以作为参数嵌入到其他命令行

```
[root@server0 ~]# wget  http://classroom/pub/materials/userlist
.. ..
[root@server0 ~]# cat  /root/userlist
duanwu
zhongqiu
zhsan
lisi
[root@server0 ~]# echo  $(cat  /root/userlist)
duanwu zhongqiu zhsan lisi
```

案例5：编写一个批量添加用户脚本

Tedu.cn 达内教育

课堂练习

在 server0 上创建 /root/batchusers 脚本

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）用户列表测试文件：
http://classroom/pub/materials/userlist

总结和答疑

脚本语法错误

问题现象

知识讲解

- 编写的 /root/foo.sh 脚本运行异常
  - 问题1：若提供参数则正常，否则会报错 line xx : [: =: unary operator expected
  - 问题2：执行脚本时报错 line xx: [: missing `]'

```
[root@server0 ~]# /root/foo.sh
/root/foo.sh: line 2: [: =: unary operator expected
.. ..
[root@server0 ~]# /root/foo.sh
/root/foo.sh: line 2: [: missing `]'
.. ..
```

故障分析及排除

Tedu.cn 达内教育

知识讲解

- 原因分析
  - 问题1：[ ] 测试操作缺少必要的参数，很可能是参与运算的变量未加双引号，当变量为空值时会报错
  - 问题2：若 [ ] 操作符与参数之间无空格则语法有错误
- 解决办法
  - 问题1：添加双引号，比如改成 if [$1 = "redhat"]
  - 问题2：[参数1 .. .. 参数2]