天津医科大学理论课教案首页

课程名称：Linux 系统概论　　**课程内容/章节：**shell 编程 / 第 13 & 14 章

教师姓名：伊现富　　**职称：**讲师　　**教学日期：**2019 年 6 月 18&25 日 13:30-15:10

授课对象：生物医学工程与技术学院 2017 级生信班（本）　　**听课人数：**28

授课方式：理论讲授　　**学时数：**4　　**教材版本：**Unix 入门经典，第 1 版

教学目的与要求（分掌握、熟悉、了解、自学四个层次）：

- 掌握 shell 脚本的结构及其运行方法，shell 脚本的注释、变量、特殊变量以及退出状态，shell 脚本中流程控制的语法，shell 脚本中函数的使用，常用的文件测试和比较运算符。
- 熟悉 shell 脚本从键盘读取输入的方法，输入输出重定向和命令替换的方法，脚本的调试。
- 了解 shell 函数的嵌套和递归、作用域。
- 自学 shell 脚本中数组的使用。

授课内容及学时分配：

- (5') 引言与导入：总结 shell 的职责，简单介绍 shell 脚本。
- (25') 编程起步：讲解 shell 脚本的结构、运行方法、注释、变量使用、特殊变量和退出状态等基础知识。
- (20') 流程控制：介绍流程控制的分类，讲解 if-then、case 等条件流程控制的逻辑流程和语法结构，讲解 while、until、for 等迭代流程控制的逻辑流程和语法结构，讲解常用的文件测试和比较运算符。
- (10') 高级概念：总结输入输出重定向和命令替换的基本方法。
- (30') shell 函数：讲解 shell 函数的声明与调用、返回值、嵌套和递归、作用域等相关知识，介绍数组的使用。
- (5') 脚本调试：介绍对 shell 脚本进行调试的方法。
- (5') 总结与答疑：总结授课内容中的知识点与技能，解答学生疑问。

教学重点、难点及解决策略：

- 重点：shell 脚本的结构、运行方法、变量等基础知识，shell 脚本中的流程控制。
- 难点：shell 脚本中的流程控制，shell 函数的声明与调用、返回值、作用域等相关知识。
- 解决策略：通过实例分析帮助学生理解记忆，结合逻辑流程和语法结构讲解流程控制。

专业外语词汇或术语：

退出状态（exit status）　　条件流程控制（conditional flow control）

流程控制（flow control）　　迭代流程控制（iterative flow control）

辅助教学情况：

- 多媒体：shell 脚本的结构、特殊变量和退出状态，各种流程控制的逻辑流程，shell 函数。
- 板书：shell 脚本中各种流程控制的语法结构。
- 演示：shell 脚本及函数实例。

复习思考题：

- 如何给变量赋值、访问变量的值？
- 列举常用的特殊变量并解释其含义。
- shell 的退出状态有哪些？
- 各种条件和迭代流程控制的语法结构。
- 列举常见的文件测试并解释其含义。
- 字符串和整数值比较的运算符有哪些？
- 如何声明并调用 shell 函数？
- 如何对 shell 脚本进行调试？

参考资料：

- (美) Harley Hahn 著，张杰良 译。Unix & Linux 大学教程，清华大学出版社，2010。
- 鸟哥 著，王世江 改编。鸟哥的 Linux 私房菜——基础学习篇（第三版），人民邮电出版社，2010。
- 维基百科等网络资源。

主任签字：　　**年　月　日**　　**教务处制**

一、 引言与导入（5 分钟）

1. shell 的职责：执行命令、管道连接、变量和文件名替换、IO 重定向、编程语言解释、环境控制……
2. shell 脚本：即 shell 程序，存储在单个文件中的一系列系统命令（实质上是一组按顺序执行的命令）

二、 【重点】编程起步（25 分钟） （实例分析、操作演示）

1. 脚本结构与运行方法

(1) 脚本结构

- #!（shebang 结构）调用 shell
- # 进行注释
- 命令和控制结构

(2) 运行方法

- 创建文件：vim script.sh
- 修改权限：chmod u+x script.sh
- 运行脚本：./script.sh，sh script.sh

2. 脚本注释

(1) 注释格式

- 注释不被解释为命令
- 注释行以 # 开头
- 多行注释的每行开头都要有 #
- 行中间可以插入 # 添加注释

(2) 注释内容

- 脚本的基本信息
- 脚本的修改日志
- 脚本每个部分的作用
- 由用户添加的数据

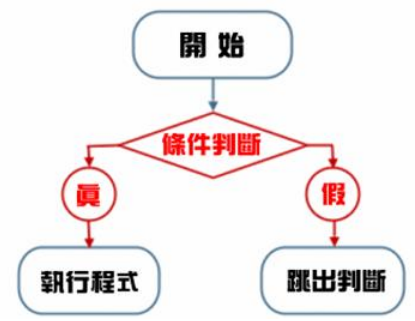
3. 变量使用
   - 赋值：使用 =（赋值运算符），= 两边不能有空格，在 = 后面跟一个换行符赋空值
   - 访问：默认情况下将变量视作文本字符串，在变量名前加 $ 可以访问变量的值
   - 变量名：只能包含字母、数字和下划线，必须以字母或下划线开头，大小写敏感（惯例使用大写）

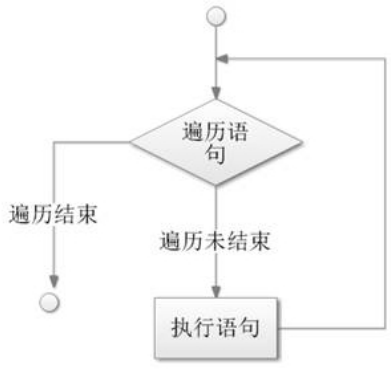
内置变量		含义
	$?	最后一次执行的命令的返回码
	$$	shell进程自己的PID
	$!	shell进程最近启动的后台进程的PID
位置变量	$#	命令行参数的个数(不包括脚本文件的名字在内)
	$0	脚本文件本身的名字
	$1 $2,...	第一个,第二个,....,命令行参数
	"$*"	"$1 $2 $3 $4...",将所有命令行参数组织成一个整体,作为一个单词
	"$@"	"$1" "$2" "$3"...,将多个命令行参数看作是多个"单词"

4. 从键盘读取输入：使用 read 读取键盘输入并为变量赋值
5. 特殊变量
6. 退出状态：0 表示成功，1 表示不成功

三、 【重点、难点】流程控制（20 分钟） （结合逻辑流程讲解语法结构，并进行实例分析）

1. 简介
   - 流程控制允许程序做出判断：程序计算条件的值，并根据这些条件执行相应的操作
   - 条件流程控制：根据特定的约束是否满足来决定是否执行某个代码段
   - 迭代流程控制：代码块重复或迭代，直到满足某个条件为止





2. 条件流程控制

- if-then
  - if-then

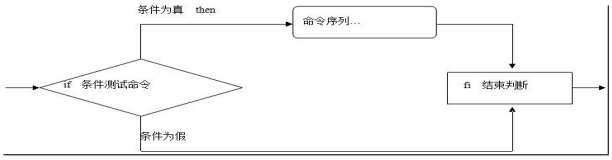
```
if some_condition
then
  something happens
fi
```

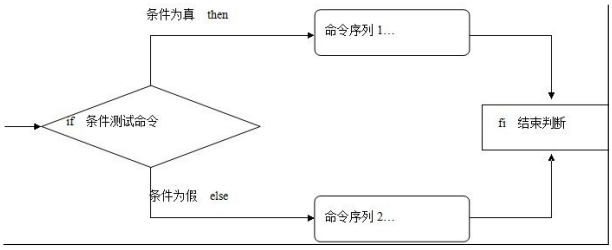
  - if-then-else

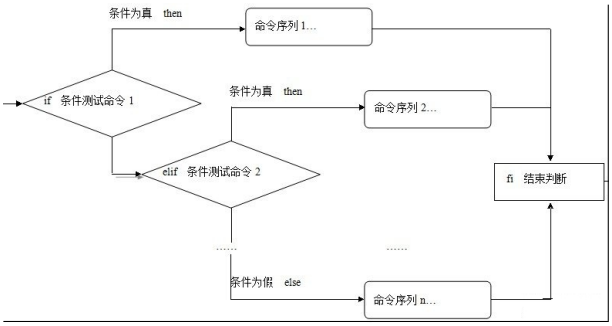
```
if some_condition
then
  something happens
else
  something happens
fi
```

  - if-then-elif-else

```
if some_condition
then
  something happens
elif other_condition
  something happens
else
  something happens
fi
```







- test
  - 语法

```
# 注意中括号内的空格
if [ $COLOR="purple" ]
if (test $COLOR="purple")
# 测试文件是否存在
if [ -e filename ]
if (test -e filename)
```

  - 文件测试

测试	助记	功能
-d	Directory	文件存在并且是目录
-e	Exist	指定的文件存在
-f	File	文在存在并且是普通文件
-G	Group	执行命令的用户属于文件所属组的成员
-nt	Newer Than	file1 -nt file2, 前一个文件比后一个文件新
-ot	Older Than	file1 -ot file2, 前一个文件比后一个文件老
-O	Owner	执行命令的用户是文件的所有者
-r	Read	执行命令的用户拥有对文件的读取权限
-s	Size	文在存在并且不为空
-w	Write	执行命令的用户拥有对文件的写入权限
-x	eXecute	执行命令的用户拥有对文件的执行权限

- 比较运算符
  - 字符串比较
  - 整数值比较

运算符	示例	功能
=	stringA=stringB	stringA 与 stringB 相同
!=	stringA!=stringB	stringA 与 stringB 不同
>	stringA>stringB	stringA 大于 stringB
<	stringA<stringB	stringA 小于 stringB
-z	-z string	string 为空
-n	-n string	string 非空

- 多个条件
  - 逻辑运算符 and（&&）

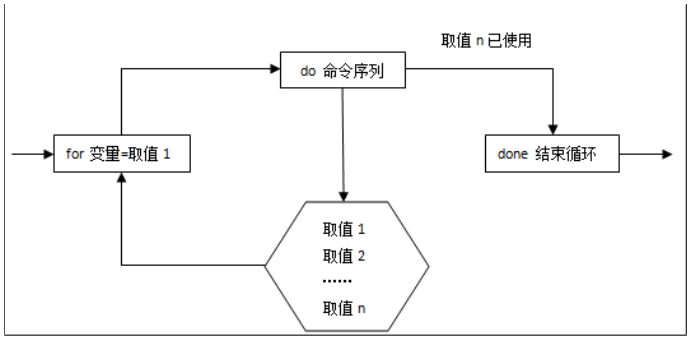
```
if [ cond1 && cond2 ]
then
  some action
fi
```

```
# 等价于
if [ condition1 ]
then
  if [ condition2 ]
  then
    some action
  fi
fi
```

运算符	助记	功能
-eq	EQual	等于
-ne	Not Equal	不等于
-gt	Greater Than	大于
-lt	Lesser Than	小于
-ge	Greater or Equal	大于等于
-le	Lesser or Equal	小于等于

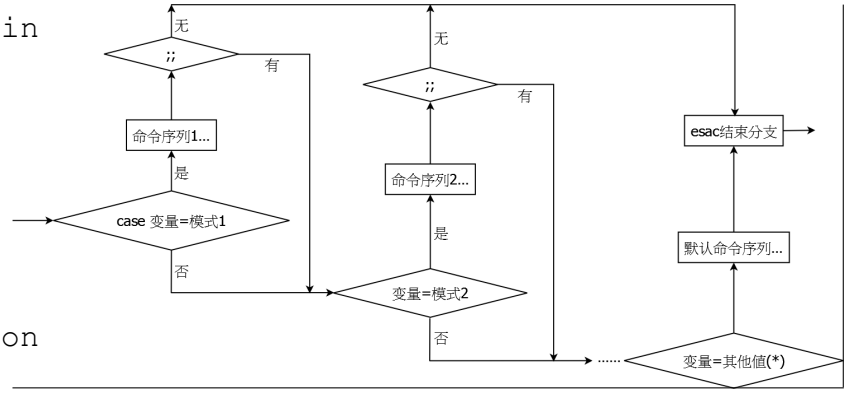
– 逻辑运算符 or（||）

```
if [ cond1 || cond2 ]
then
  some action
fi
# 等价于
if [ condition1 ]
then
  some action
elif [ condition2 ]
then
  the same action
fi
```



- case

```
case expression in
  pattern1)
    action1
    ;;
  pattern2)
    action2
    ;;
  *)
    default action
    ;;
esac
```



3. 迭代流程控制
   - while

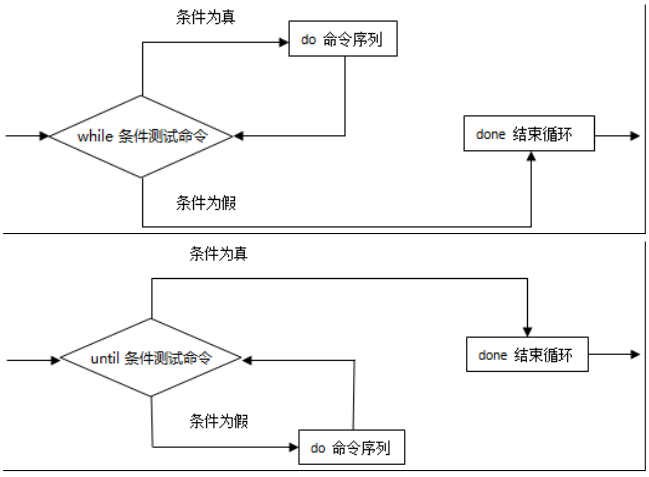
```
while condition
do
  action
done
```

   - until

```
until condition
do
  action
done
```

   - for

```
for VAR in LIST
do
  action
done
```



类别	操作符	说明
输入重定向	<	输入重定向是将命令中接收输入的途径由默认的键盘更改（重定向）为指定的文件
输出重定向	>	将命令的执行结果重定向输出到指定的文件中，命令进行输出重定向后执行结果将不显示在屏幕上
	>>	将命令执行的结果重定向并追加到指定文件的末尾保存
错误重定向	2>	清空指定文件的内容，并保存标准错误输出的内容到指定文件中
	2>>	向指定文件中追加命令的错误输出，而不覆盖文件中的原有内容
输出与错误组合重定向	&>	将标准输出与错误输出的内容全部重定向到指定文件

四、 高级概念（10 分钟）

1. 输入输出重定向
2. 命令替换
   - ``
   - $()

五、 【难点】shell 函数（30 分钟） （实例分析、操作演示）

1. 声明与调用

```
# 函数声明
name () {
  commands
}
# 函数调用，不带()
name
name parameter(s)
```

```
#!/bin/bash
# A simple function
repeat () {
  echo "I don't know $1 $2"
}
repeat Your Name
# I don't know Your Name
```

2. 返回值：默认是最后一条语句的状态，可以使用 return 显式地设置函数的退出状态
3. 嵌套和递归
   - 递归函数：就像调用其他函数一样、调用自己的函数
   - 函数嵌套：在一个函数中调用包含在另一个函数中的功能
4. 文件处理

```
if [ -r File && -x File  ]
then
  echo "File exists, is readable, and executable!"
fi
if [ -r File || -w File  ]
then
  echo "File exists and is readable or writable!"
fi
```

5. 作用域
   - 全局作用域：在脚本的任何地方都可以访问变量
   - 局部作用域：只能在声明变量的作用域内访问它们
   - 使用 local 关键字定义局部变量
6. 数组
   - 声明
     - `array[index]=value`
     - `array2=(value1 value2 value3 ...)`
     - `array3=([0]=value1 [13]=value2 [7]=value3)`
   - 解引用数组
     - 获得数组中某个特定索引位置的数据：`${array[index]}`
     - 获得数组中的所有数据：`arrayElements=${array[@]}`
     - 获得数组包含的元素数目：`arrayLength=${#array[@]}`
     - 获得数组中包含的索引值：`arrayIndex=${!array[@]}`
     - 获得第四个元素（含）之后的所有元素的数据：`afterTue=${day[@]:3}`
     - 获得第四个元素（含）后面两个元素的数据：`WedThu=${day[@]:3:2}`
   - 删除数组
     - 删除索引位置为 1 的元素的数据：`unset array[1]`
     - 删除数组中所有元素的数据：`unset array[@]`

六、 脚本调试（5 分钟）

- `sh -n script`：检查语法，不执行脚本。
- `sh -x script`：执行脚本并显示所有变量的值。
- `set -x; command; set +x`：脚本局部调试。
- `echo`：打印变量的值、提示信息等来调剂脚本。

七、 总结与答疑（5 分钟）

1. 知识点
   - shell 脚本编程起步：脚本结构、运行、注释，调用 shell，变量、特殊变量，从键盘读取输入，退出状态
   - 条件流程控制：if-then，test，case
   - 迭代流程控制：while，until，for
   - shell 编程高级概念：输入输出重定向，命令替换
   - shell 函数：声明与调用，返回值，嵌套和递归，作用域，文件处理，数组
   - shell 脚本的调试
2. 技能
   - 掌握 shell 编程的基本语法和调试命令
   - 使用 shell 编写应用脚本