Chapter 6

Conditional Statements

Overview

- The exit status of commands
- The `test` command
- The `if` construct
- `case` statement

Lesson: The exit status of commands

- Every sub-process will return exit code to calling process
- This code indicates the exit status of ended process
- It is assumed that:
  - Exit code 0 means success
  - Everything else means error
  - Specific exit codes have specific meaning (depending on process).

The Return Status Variable

- Sub-process exit code is available through special environment variable $?
- Every command will change $? Value
- Script exit code is taken from last command

```
$ cp mamo memo2
cp: cannot find file "mamo"
$ echo $?    # result of cp failure
2
$ echo $?    # result of previous echo command
0

$ cp memo memo2
$ echo $?    # result of successful copy
0
```

The `exit` Command

- Syntax

```
exit [n]
```

- The script ceases as soon as the **`exit`** command is given.
- The **`exit`** command exits the script using the current value of **`$?`**
- If ***n*** is specified, the exit status will be set to ***n***.

Lesson: The `test` command

- `test` – evaluate expression
- `test` – operators
- Multiple `test` Conditions

`test` – evaluate expression

- **Syntax**

```
test [ expression ]
[ [ expression ] ]
```

  - The `test` command evaluates the expression, and sets the return code.

Expression value	Return code
True	0
False	Non-zero (commonly 1)

- **The `test` command can evaluate the condition of**
  - Integers
  - Strings
  - Files

`test` – operators

Binary operators		Unary operators	
String tests			
`=`	Equal	`-z`	length = 0
`!=`	Not equal	`-n`	lenght > 0
Numeric tests		**File tests**	
`-eq`	Equal	`-r`	Has read permission
`-ne`	Not equal	`-w`	Has write permission
`-lt`	Less than	`-x`	Has execute permission
`-le`	Less or equal	`-f`	Is file
`-gt`	Greater than	`-d`	Is directory
`-ge`	Greater or equal	`-s`	Size > 0
Logical operators			
`-a`	And	`!`	Not
`-o`	Or		

Multiple `test` Conditions

```
[ -f $var ] && [ -r $var ] && [ -w $var ]
```

Is it a file? AND is it readable? AND is it writable?

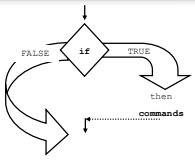
Lesson: The `if` construct

- `if`
- `if-else`
- More about `if`
- Compound `if`

The `if` construct

- Syntax

```
if
    list A
then
    list B
fi
```



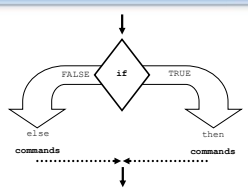
- Example

```
if
  test -s funfile
then
  echo funfile exists
fi
echo hello
```

The `if-else` construct

- Syntax

```
if
    list A
then
    list B
else
    list C
fi
```



- Example

```
if [ "$X" -lt 10 ]
then
  echo X is less than 10
else
  echo X is not less than 10
fi
```

More about `if`

- **Nested `if`**

```
if [ condition_1 ]
then
    if [ condition_2 ]
    then
        commands_1
    else
        commands_2
    fi
else
    commands_3
fi
```

Nested `if`

- **Shell command lists**

```
command_1 && command_2                # if then
command_1 || command_2                # if else
command_1 && command_2 || command_3   # if then else
```

Compound `if` — Multiway Decision Branching

- **Syntax**

```
if   [ condition_1 ]
then
   commands_1
elif [ condition_2 ]
then
   commands_2
elif [ condition_3 ]
then
   commands_3
else
   commands_4
fi
```

- **Example**

```
if [ -d oldA ]
then
  echo "Moving memo to oldA"
  mv memo oldA
elif [ -d oldB ] ; then
  echo "Moving memo to oldB"
  mv memo oldB
elif [ -d oldC ] ; then
  echo "Moving memo to oldC"
  mv memo oldC
else
  echo "No dir exist."
  echo "no moving occurs"
fi
```

Lesson: The `case` construct

- `case`
- **Pattern examples of `case`**

The case construct

- Syntax

```
case word in
  pattern1)  list_A ;;
  pattern2)  list_B ;;
  patternN)  list_N ;;
esac
```

- Example

```
case $ANS in
  yes)  echo O.K.
        ;;
   no)  echo no go
        ;;
esac
```

```
case $OPT in
    1)  echo option 1      ;;
    2)  echo option 2      ;;
  3|4)  echo option 3 or 4 ;;
    *)  echo no option     ;;
esac
```

The case construct — pattern examples

- The case construct patterns use the same special characters that are used to generate file names.

```
$ cat menu_with_case
clear
echo "                  COMMAND MENU\n"
echo "    d  to display time and date"
echo "    w  to display logged-in users"
echo "    l  to list contents of current directory"
echo "       Please enter your choice : \c"
read choice
case $choice in
  [dD]*)    date ;;
  [wW]*)    who  ;;
  l*|L*)    ls   ;;
      *)    echo Invalid selection ;;
esac
$
```

Review Exercises

- Complete the exercises from the Learning Guide

Topics for Review