Testing Conditions in Scripts

1 Testing Conditions with the If Command

- if (expression) command

```
if ( ! $?var1 ) set var1 = defineitnow
```

- if (expression) then

    command1

    command2

  endif

```
if ( ! $?var1 ) then
   set var1 = defineitnow
   echo "The variable 'var1' is now: $var1"
endif
```

- There is an `else` version too.

```
if (! $?var1) then
   set var1 = defineitnow
   echo "The variable 'var1' is: $var1"
else if ($var1 == "definediteearlier") then
   echo "The variable 'var1' was: $var1"
else
   set var1 = makeitanewdefinition
   echo "The variable 'var1' is now: $var1"
endif
```

2 Using Switch to Create Case Analysis

This statement has the following form:

```
switch (string)
case str1:
   command1
   command2
   breaksw
case str2:
   command3
   breaksw
default:
   command4
endsw
```

For example:

```
switch (`tty`)
case /dev/ttyp*:
   # terminal type is a pseudo terminal
   setenv TERM xterm
   breaksw
case /devtty8:
case /devtty9:
   # terminal type is a hardwired type
   setenv TERM vt100
   breaksw
case default:
   # terminal type unknown
   set out = `tty`
   echo "tty $out is unrecognized"
endsw
```

- `tty` is a unix command.

```
% tty
/dev/ttyp1
```

- The **backquoted** string `` `tty` `` in the `switch` statement is executed.

  The result becomes the string tested by the `case` statements.

```
switch (`hostname`)
case spooner:
   # host is on NeXT fileserver
   set local = /usr/ucb
   breaksw
case garfield:
case herman:
   # host is a new sparc
   set local = /usr/local
   breaksw
case apple:
case pear:
   # host is a sun 3/50
   set local = /usr/uts
case default:
   # host type unknown
   set local = /usr
   echo "host is unrecognized"
endsw
set path = ($local /usr/bin /bin /usr/support )
```