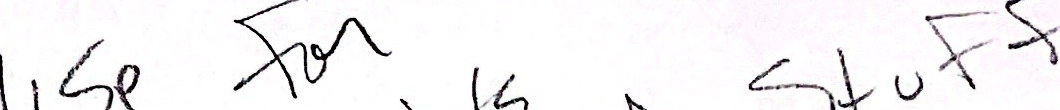
**9**

**Exit/Test**



Exit/Test

**Objectives**

Use the test command to **check** **fie** **status,** to **compare** strings, and to do numerical comparisons

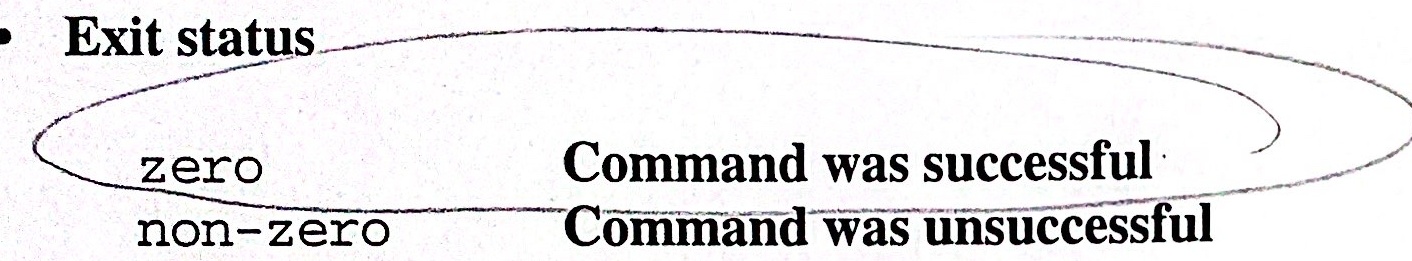
Use the exit statement **for** **premature** **termination** **or** **to** terminate with a specified exit status.

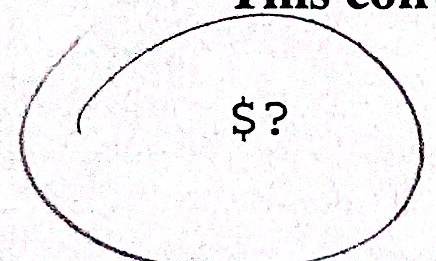
·Form compound conditional **statements using** and **and** or



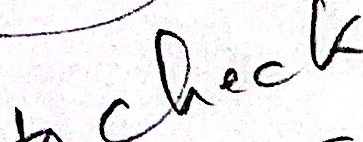
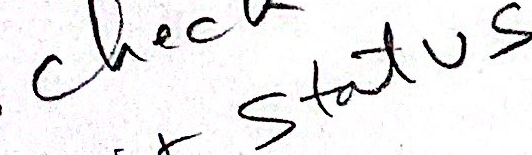
Notes

**exit Status**



This **convention** is opposite **to** that used **in** **C** programming.

**Stores** exit status of lastcommand executed.

**Notes**

Create a command file and assign a nonzero exit.

$ cat > nonzero

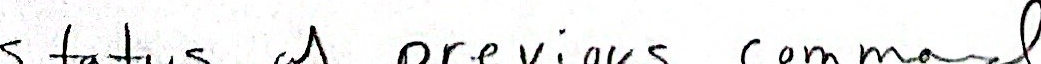
date

exit 2

Control-D

$ sh nonzero

<Displays date>

$echo 

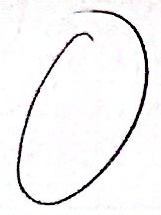
2

$ echo 

0 

UNIX Shell Programming

©Copyright 1996-2000 GTRC, All Rights Reserved



**exit Status (Cont'd)**

·Failures: (Non-Zero exit status)

Failures can be caused by:

1. invalid arguments passed to the program

2. an error condition detected by the program

3. program failed to match pattern

Example**:**

grep "pattern" \*

**Exit Status** **Condition**

0 **Pattern matched**

1 **Pattern not found**

2 **Error in pattern or command options**

**Notes**

If several commands are grouped together using parentheses,the exit status is the exit status of the last enclosed executed command.

Non-zero exit status values may vary from command to command.



Exit/Test

**The test Command**

**· Format:**

test expression

**or**

[ expression]

**· Arguments:**

**"ands" criteria**

-a

**"ors" criteria**

-0

**negates criteria**

!

**groups criteria**

()

**Notes**

The test command is used by Shell programs.

The arguments to test form an expression. test returns a zero value if the test was successful and a non-zero value otherwise.

· If no arguments are given, test returns a nonzero exit status.

· Shell variables should be enclosed in double quotes if they are null or not set.

·Spaces are necessary around square brackets.

·The test expression is equivalent to [ expression ].

·Each argument to test is delimited by spaces.

9-5

UNIX Shell Programming

©Copyright 1996-2000 GTRC,All Rights Reserved

**Types of Tests**

·There are three types of tests:

1. Tests on numerical values

2. Tests on file types

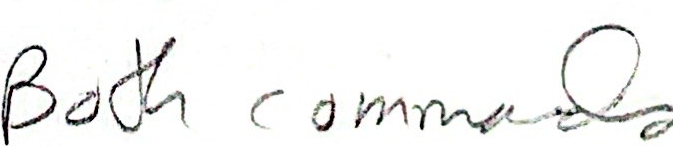
3. Tests on character strings

**Notes**

testis equivalent to[ ]

**test** -f /csri/rice/.login

or

[ -f /csri/rice/.login ]

All primitives, operators, and file names are separate arguments to test.

**Tests on Numerical Values**

· N <primitive> M **(M and N are integers)**

**The primitives include:**

-eq **equal (N=M)**

-ne **not equal** (N≠M)

-gt **greater than** (N>M)

-1t **less than** (N<M)

-ge **greater than or equal to** (N>=M)

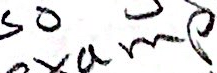
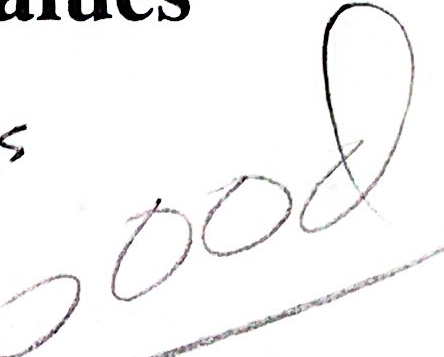
-1e **less than or equal to** (N<=M)

**Notes**

test "$number" -eq 0 tests the variable number to see if it is equal to zero.

test "$number" -gt "$max" tests the variables number and max to determine if number is greater than max.

**Examples Testing Numerical Values**

**· Example 1:**

users=`rwho| **wc** -1`

**if** **test** "$users" **-gt** 10

**then**

echo "the network is busy"

**fi**

**Example 2:**

number1=1

number2='1'

**if** **test** "$number1" **-eq** "$number2"

**then**

echo "$number1 is equal to $number2"

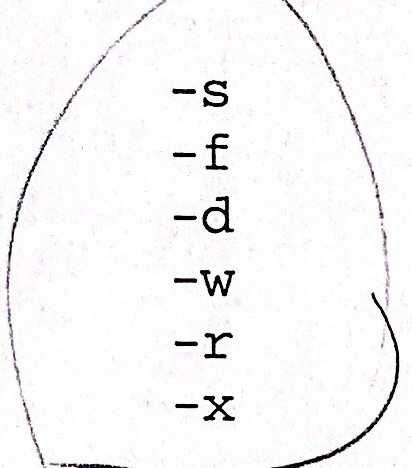
fi



**Notes**

**Tests on File Types**

· <primitive> filename

**The following tests return true if**

**the file exists and is not empty**

**the file exists and is not a directory**

**the file exists and is a directory**

the **file exists and is writable**

**the file exists and is readable**

thefile **exists and is executable**

!<primitive> **inverts the sense of the primitive.**

**Notes**

·Each of the above operators is a unary operator, thatis the operator expects only one argument.

! <primitive> must be surrounded by spaces.

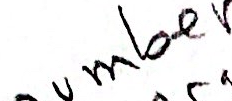


test -f filename The file existsand is not.empty

test ! -d filename The file does not exist or it is not a directory

test ! -r filename The file does not exist or it is not readable

**Examples Testing File Types**

**·Example 1:**

**if** **test** $# -eq 0

**then**

echo "You did not enter an argument"

exit 1

**elif** test -f $1 -a -r $1

**then**

cat $1

**elif test** -d$1

**then**

echo "the file is a directory"

**else** echo "it must be a special file

or does not exist"

**fi**

**· Example 2:**

**if** **test** $# -eq 0

then

echo You did not enter a filename

**elif test** !-s$1

**then**

echo The file does not exist or it is empty

**else** cat $1

**fi**



**Notes**

**Examples Testing File Types(Cont'd)**

**·Example 3:**

[ -s /usr/tom/memo ]

**tests file** memo **to see if it contains at least one byte of information**

**· Example 4:**

[ ! -r /usr/tom/memo ]

**tests file** memo **to see if it does not exist or it is not readable**

**Example 5:**

[ ! -f /usr/tom/memo ]

**tests file** memo **to see if it does not exist or is not an ordinary file**

**Example 6:**

[ ! -x /usr/tom/memo]

**tests file** memo **to see if it does not exist or it is not executable**

**Notes**

Notice the explicit use of spaces around all operators and arguments. Also there must be spaces around the brackets.



Exit/Test

**Tests on Character Strings**

· S1 <primitive> S2

**There are two binary string comparison primitives:**

= **The strings are equal**

!= **The strings are not equal**

**Notice there is no space between the ! and =.**

<primitive> S

**There are two unary string comparison primitives.**

-Z **String S has zero length**

-n **String S has non-zero length**



**Notes**

month=nov Value

tes true

test false

test1 false

month=""

test -n $month

is equivalent to test $month. If $month is not null, then true is returned.

**Examples Testing Character Strings**

**Example 1:**

$ string1=hi

$ string2=jane

**$** **test** "$string1"= "$string2"

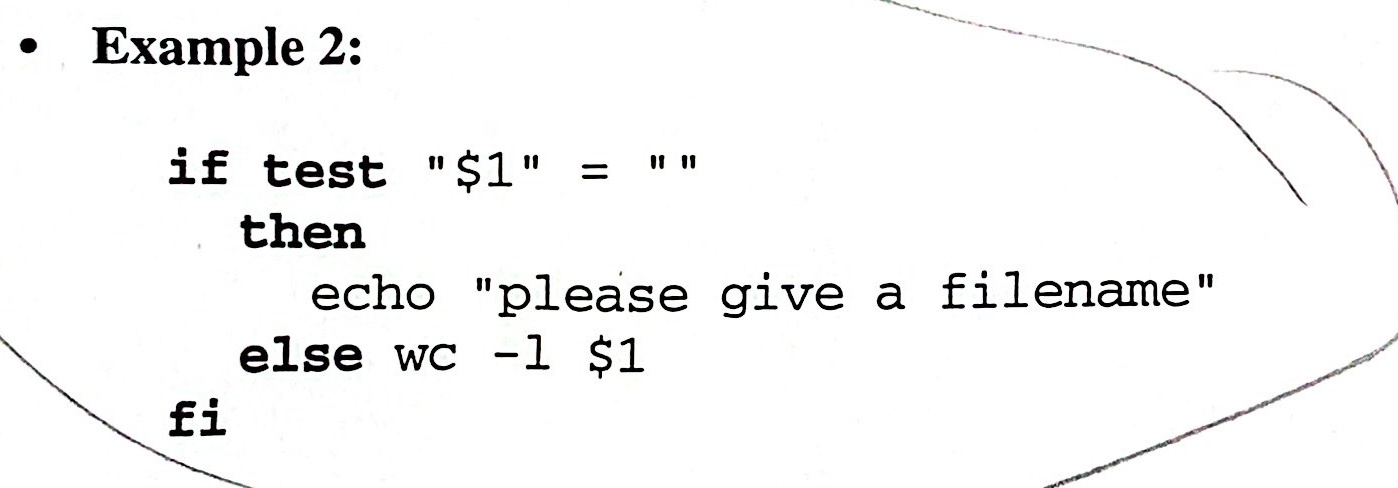
$ echo $? #returns a nonzero exit

1 #status

$ **test** "$string1"!= "$string2"

$ echo $? #returns a zero exit status

0



**Example 3:**

**if** **test** -z "$1"

**then**

echo "please give a filename"

**else** wc -1 $1

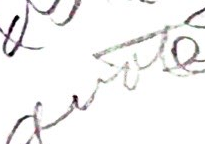
**fi**

**Notes**

**Examples Testing Character Strings(Cont'd)**

**· Example 4:**

#true



1 #false

$

0 #true



**Notes**

·If the double quotes are left off of $day, the Shell does not see the extra space after monday.

**Comparison of Character Strings and**

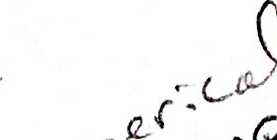
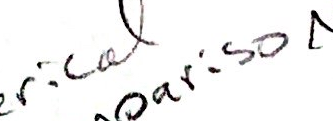
**Numeric Strings**



string comparison

$

1

$  integer comparison

$echo $

0

$ [ $n1 -eq 11 ] integer comparison

$ echo $?

0

$  -eq 11 ] integer comparison

Secho

0

**Notes**

**Logical AND and OR Operators**

AND -a

expr1 **-a** expr2

**returns true only if expr1 and expr2 are both true.**

**·Example 1:**

**[** **-f** /usr/tom/memo1 **-a** **-r** /usr/tom/memo2 **]**

**tests** memo1 **to see if it is an ordinary file and** memo2 **to see if it is** **readable.**

**Example 2:**

[ "$count" **-ge** **5** **-a** "$count" **-1e** 15 ]

**tests to see if** count **is greater than or equal to** 5 **and if** count **is less** **than or equal to** 15.



**Notes**

**Logical AND and OR Operators (Cont'd)**

OR -O

expr1 -o expr2

**returns true if expr1 is true or if expr1 is false and expr2 is true.**

**· Example 1:**

[ **-s** /usr/tom/letter **-o** **-r** /usr/tom/doc ]

**tests 1etter to see if it is not empty. If it is not, then true is returned. If** letter **is empty, then doc is tested to see if it is readable.**

**· Example 2:**

["$a"**-eq** 0 **-o** "$b" **-gt** 10 **-a "$b" -1t** 20 **]**

**returns true if $a equals zero or if $b is bounded between 10 and 20.The arguments of -a are evaluated and a value is returned. That value is** **then passed as an argument to -o.**

**Notes**

Parentheses may be used to change the order of operation. In many instances, the parentheses must be quoted since they have a special meaning to the Shell,\(or\).

· -a has a higher order of precedence than-o.

· -o is evaluated from left to right.

**Summary**

The exit status of a program can be used to control execution of shell conditional constructs.



An exit status of zero means success. Non-zero means failure.

$? has the exit status of the last command executed.

The test command, and its synonym [...] have a number of operations.They can

- compare numbers

- compare strings

- test file attributes

-combine test criteria with “and”, “or”, and “not” operations

Double

The use of quotes with variables as arguments to test is important, and can affect the results of the test.



Notes

**10**

**Conditional Constructs**