J.MOHAMED THAMEEM M.Sc (IT), M.Phil(CS).,
PROGRAMMER,
DEPARTMENT OF COMPUTER APPLICATIONS
JAMAL MOHAMED COLLEGE (AUTONOMOUS)

TRICHY

Shell Meta Characters
Redirection characters

- < or 0< redirect for standard input from a specified file eg: command < filename
- > or 1> redirect standard output into the specified file eg: command > filename
- 2> redirect standard error into the specified file eg: command 2> filename
- >> append standard output into the file eg: command >> filename
- < take standard input from the specified file

  eg: command < filename

| connect standard output of one command (c1) into the standard input of another command(c2) eg: c1| c2

Pattern matching characters

- ? - matches any single character in file name

Eg: a?d represent 3 chars filename starting with ‘a’ ending with ‘d’ and middle may be any character

- - matches any string of zero or more characters in filename. Eg: a * c represent filename starting with ‘a’ ending with ‘c’ and middle may be combination of character of any length
- [character_list] - matches any single specified character in filename eg: a[bc]d represent the filename abc or acd.
- [c1-c2] - matches a single character that is within the ASCII range of character c1 and c2.
- [character_list] – matches any single character that is not specified in character_list.

Command terminating character

; - separates the command when more than one command is given in line.

Eg: command 1 ; command 2

Executed command1 and then execute command 2

& - like ; character, but does not wait the previous command to complete.

Eg: command1 & command2

Unlike like the ; character, the command 2 will not wait for the completion of command 1

Comment character

- if an # symbol appears in a line, then the rest of the line will be treated as comment.

Eg: #this is a comment

Conditional execution character

&& - command1 && command2

execute command1, if successful execute command 2

|| - command1 || command2

Execute command1, if failure execute command2

ITERACTIVE STATEMENTS

- These statements are used to execute a sequence of statement based on conditions

1. While loop

g.f: while <conditional_command>

do

<commands>

done

The <conditional _ command> is executed for each cycle of loop. If it return a zero exit status.

ITERACTIVE STATEMENTS

2. Until loop

G.f : until <conditional command>

do

<command>

Done

This loop is similar to while loop except that it continues as long as the

< conditional_command> fails.

ITERACTIVE STATEMENT

3. FOR LOOP

g.f: for <variable_name> in <list_of_values>

Do

<commands>

Done

The <variable_name> has a value from <list_of_values> in each cycle of loop. And the loop will be executed until the <list_of_values> becomes empty.

ITERACTIVE STATEMENTS

4) Break

This command is used to exit the enclosing loop or case command. The optional parameter n is an integer value that represents the number of levels to break when loop command are nested. This method is very convenient to exit a deeply nested looping structure when some type of error has occurred.

g.f: break [n]

Where n represent the number of levels to break.

ITERACTIVE STATEMENTS

5) Continue

This command is used to skip to the top of the next iteration of looping statement .

Any command within enclosing loop that follow this continue statement are skipped and the execution continues at top of the loop. If the optional parameter “n” is used then the specified number of enclosing loop levels are skipped.