**DAY – 03**

**SHELL VARIABLES:**

System variables-------boot, login

ECHO: It acts as a print statement

Ex: $A=10

Echo “hello world”

Echo $A

Out:10

o echo $PS1—profile(primary prompt string)

o echo $SHELL

o history—command history that we have done

Shell Scripts Commands: vi. Bashrc

o vi myscript.sh--creating a shell script

o sh .myscript.sh—we can execute without the execute permission

o ./myscript.sh—execute in command prompt

**ACCEPT NAME & DISPLAY:**

echo “Welcome to Shell Scripting”

echo “Enter the Name:”

read name

echo “Entered Name is:$name”

ADDITION OF TWO NUMBERS:

echo "Enter the first number:"

read num1

echo "Enter the second number:"

read num2

num3=`expr $num1 + $num2`

echo "Sum is: $num3"

PROGRAM:

var=pwd

echo $var

var=`pwd`

echo $var

var=`date`

echo $var

· When we pass command line arguments in shell script it comes under special variables.

· These are called positional parameters.

· $\*--name of all the arguments passed

· $#--number of arguments passed

· $$--Gives the PID of the current shell

· $!—Gives the PID of the last background job

· $?—Gives the exit status of last command

· $@-- Similar to $\*, but generally used with strings in looping constructs

PROGRAM:

echo program:$0

echo "number of args"

echo $#

echo "all the args "

echo $\*

echo "last args:"

echo $1

· wc—It gives number of words, lines, characters in a particular file.

· find—used to find the file in a directory

o By default, find does not allow symnollic links

o Syntax: find pathname\_list[expression]

­ find ./--name test1.txt

­ find ./--name \* test1.txt

­ find ./--name “\*.cpp”

­ find ./--name “t\*.\*”

­ find ./--name “\*1.\*”

**FILTER**: Filter is a command that takes its input from standard input, processes it and sends its output to the standard input, processes it and sends its output to the standard output.

o Commands such as ls, date, pwd etc. can not be used as filters as they do not require any input.

o Commonly used filters: grep, sort, cut, paste, head, tail, wc, pg, more, tr.

v grep—Global Search for regular expression & print untility.

o Used to search for a particular pattern of characters and display all the lines containing the pattern.

o $grep[option]<pattern><file name>

o Options: -n:prints line numbers

-r:the reverse search criterion

-c: display only a count of matching patterns

o Regular Expression:

­ “^” :beginning of line

­ “$”:end of line

­ “.” :any single character

­ […] :any one character from the list

­ [^…] :String beginning

o Options with grep:

­ -v: display those lines that do not match

­ -n: precede each matching line with the line number

­ -e: print only the total count of matched lines

­ -i: ignore cases

­ -l: lines that matched the pattern

o $grep “text..” file name—without opening file check the data is present or not

o $grep “^B” employee dat--it will display all lines in filename that begin with the letter "B".

o $grep “S$” employee.dat--it will display all lines in filename that end with the letter "S".

o $grep “B^” employee.dat—it will display all lines in file that contain the substring “ B^”.

o $grep -i “ ^n” employee.dat—it will display all lines in file that start with the letter “n”, treating the search as case sensitive.

o $grep “6…$” employee.dat—it will display the number with 6 digit along with 3 characters

o $grep “B…$” employee.dat

o $grep “B…\*” employee.dat-- it will display all lines in filename that start with "B", followed by any three characters, and then possibly more characters.

o $grep “B\*” employee.dat—it will display the line that start with “B” along with all lines

o $grep “T[0-9][0-9] employee.dat--it will display all lines in filename that contain "T" followed by two digits, like "T23" or "T07".

o $grep -i ”1[0-9][0-9]” employee.dat—it will display all lines that contain any occurrence of "1" followed by two digits.

o $grep -I “B[a-z][a-z]” employee.dat—it will display all lines that contain "B" followed by two lowercase letters.

o $grep -I “90[0-9][0-9]” employee.dat—it will display all lines that contain the sequence "90" followed by two digits.

· ssh--- Secured Shell

· Putty is a telnet----telent 100.100.100.100.8080

v mkdir libs-- > cd libs -- > vi calc.c -- > gcc -c calc.c -o cal.càC file converted to binary file

**CREATING A SHARED LIBRARY:**

o Shared libraries are not embedded into the executable but are loaded at runtime. This allows multiple programs to share the same library reducing the size of executables.

o PIC---Position Independent Code