**Compute net & gross salary .make use of function**

**SOURCE CODE:**  
get()  
{  
echo “ Enter code, allowance, deduction, basic pay ”  
read code  
read all  
read ded  
read basic  
}  
display()  
{  
echo “ The code is $code ”  
echo “ The allowance is $all ”  
echo “ The deducion is $ded ”  
echo “ The basic pay is $basic ”  
}  
get $code $all $ded $basic  
gross=`expr $bacis + $all`  
net=`expr $gross - $ded`  
echo “ The gross is $gross ”  
echo “ The net is $net ”  
  
**OUTPUT:**  
Enter code, allowance, deduction, basic pay  
100  
200  
300  
400  
The gross is 600  
The net is 300

**COMPARISON OF TWO STRINGS**

**Date:**

**Aim:**

To write a shell program to compare the two strings.

**Algorithm:**

Step1: Enter into the vi editor and go to the insert mode for entering the code

Step2: Read the first string.

Step3: Read the second string

Step4: Compare the two strings using the if loop

Step5: If the condition satisfies then print that two strings are equal else print two

strings are not equal.

Step6: Enter into the escape mode for the execution of the result and verify the output

**Program:**

echo “enter the first string”

read str1

echo “enter the second string”

read str2

if [ $str1 = $str2 ]

then

echo “strings are equal”

else

echo “strings are unequal”

fi

**Sample I/P:1**

Enter first string: hai

Enter second string: hai

**Sample O/P:1**

The two strings are equal

**Sample I/P:2**

Enter first string: hai

Enter second string: cse

**Sample O/P:2**

The two strings are not equal

**Result:**

Thus the shell program to compare the two strings is executed and output is verified successfully.

**MAXIMUM OF THREE NUMBERS**

**Date:**

**Aim:**

To write a shell program to find greatest of three numbers.

**Algorithm:**

Step1: Declare the three variables.

Step2: Check if A is greater than B and C.

Step3: If so print A is greater.

Step4: Else check if B is greater than C.

Step5: If so print B is greater.

Step6: Else print C is greater.

**Program:**

echo "enter A"

read a

echo "enter B"

read b

echo "enter C"

read c

if [ $a -gt $b -a $a -gt $c ]

then

echo "A is greater"

elif [ $b -gt $a -a $b -gt $c ]

then

echo "B is greater"

else

echo "C is greater"

fi

**Sample I/P:**

Enter A:23

Enter B:45

Enter C:67

**Sample O/P:**

C is greater

**Result**:

Thus the shell program to find the maximum of three numbers is executed and output is verified successfully.

**FIBONACCI SERIES**

**Date:**

**Aim:**

To write a shell program to generate fibonacci series.

**Algorithm :**

Step 1 : Initialise a to 0 and b to 1.

Step 2 : Print the values of 'a' and 'b'.

Step 3 : Add the values of 'a' and 'b'. Store the added value in variable 'c'.

Step 4 : Print the value of 'c'.

Step 5 : Initialise 'a' to 'b' and 'b' to 'c'.

Step 6 : Repeat the steps 3,4,5 till the value of 'a' is less than 10.

**Program :**

echo enter the number

read n

a=-1

b=1

i=0

while [ $i –le $n ]

do

t=`expr $a + $b`

echo $t

a=$b

b=$t

i=`expr $i + 1

done

**Sample I/P :**

Enter the no: 5

**Sample O/P:**

0

1

1

2

3

5

**Result :**

Thus the shell program to find the fibonacci series is executed and output is verified successfully.

**ARITHMETIC OPERATIONS USING CASE**

**Date:**

**Aim:**

To write a shell program to perform the arithmetic operations using case

**Algorithm :**

Step 1 : Read the input variables and assign the value

Step 2 : Print the various arithmetic operations which we are going to perform

Step 3 : Using the case operator assign the various functions for the arithmetic

operators.

Step 4 : Check the values for all the corresponding operations.

Step 5 : Print the result and stop the execution.

.

**Program :**

echo 1.Addition

echo 2.Subraction

echo 3.Multiplication

echo 4.Division

echo enter your choice

read a

echo enter the value of b

read b

echo enter the value of c

read c

echo b is $b c is $c

case $a in

1)d=`expr $b + $c`

echo the sum is $d

;;

2)d=`expr $b - $c`

echo the difference is $d

;;

3)d=`expr $b \\* $c`

echo the product is $d

;;

4)d=`expr $b / $c`

echo the quotient is $d

;;

esac

**Sample I/P :**

1.Addition

2.Subraction

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**1: Date Command:**

This command is used to display the current data and time.

**Syntax:**

$date

**2. Calender Command :**

This command is used to display the calendar of the year or the particular month of calendar year.

**Syntax :**

a.$cal <year>

b.$cal <month> <year>

Here the first syntax gives the entire calendar for given year & the second Syntax gives the calendar of reserved month of that year.

**3. Echo Command :**

This command is used to print the arguments on the screen.

**Syntax:** $echo <text>

**Multi line echo command:**

To have the output in the same line , the following commands can be used.

**Syntax:** $echo <text\>text

To have the output in different line, the following command can be used.

**Syntax:** $echo “text

>line2

>line3”

**4.’who’ Command :**

It is used to display who are the users connected to our computer currently.

**Syntax :** $who – option‟s

**Options : -**

H–Display the output with headers.

b–Display the last booting date or time or when the system was lastely rebooted.

**6.’who am i’ Command :**

Display the details of the current working directory.

**Syntax :** $who am i

**7.’tty’ Command :**

It will display the terminal name.

**Syntax :** $tty

**8.’Binary’ Calculator Command :**

It will change the „$‟ mode and in the new mode, arithematic operations such as +,-,\*,/,%,n,sqrt(),length(),=, etc can be performed . This command is used to go to the binary calculus mode.

**Syntax :**

$bc operations

^d

$

1 base –inputbase

0 base – outputbase are used for base conversions.

Base :

Decimal = 1 Binary = 2 Octal = 8 Hexa = 16

**9.’CLEAR’ Command :**

It is used to clear the screen.

**Syntax :** $clear

**10.’MAN’ Command :**

It help us to know about the particular command and its options & working. It is like „help‟ command in windows .

**Syntax :** $man <command name>

**11.MANIPULATION Command :**

It is used to manipulate the screen.

**Syntax :** $tput <argument>

**Arguments :**

1.Clear – to clear the screen.

2.Longname – Display the complete name of the terminal.

3.SMSO – background become white and foreground become black color.

4.rmso – background become black and foreground becomes white color.

5.Cop R C – Move to the cursor position to the specified location.

6.Cols – Display the number of columns in our terminals.

**12.LIST Command :**

It is used to list all the contents in the current working directory.

**Syntax :** $ ls – options <arguments>

If the command does not contain any argument means it is working in the Current directory.

**Options :**

a– used to list all the files including the hidden files.

c– list all the files columnwise.

d- list all the directories.

m- list the files separated by commas.

p- list files include „/‟ to all the directories.

r- list the files in reverse alphabetical order.

f- list the files based on the list modification date.

x-list in column wise sorted order.

**DIRECTORY RELATED COMMANDS :**

**1.Present Working Directory Command :**

To print the complete path of the current working directory.

**Syntax :** $pwd

**2.MKDIR Command :**

To create or make a new directory in a current directory .

**Syntax :** $mkdir <directory name>

**3.CD Command :**

To change or move the directory to the mentioned directory .

**Syntax :** $cd <directory name.

**4.RMDIR Command :**

To remove a directory in the current directory & not the current directory itself.

**Syntax :** $rmdir <directory name>

**FILE RELATED COMMANDS :**

**1.CREATE A FILE :**

To create a new file in the current directory we use CAT command.

**Syntax :**

$cat > <filename.

The > symbol is redirectory we use cat command.

**2.DISPLAY A FILE :**

To display the content of file mentioned we use CAT command without „>‟ operator.

**Syntax :**

$cat <filename.

Options –s = to neglect the warning /error message.

**3.COPYING CONTENTS :**

To copy the content of one file with another. If file doesnot exist, a new file is created and if the file exists with some data then it is overwritten.

**Syntax :**

$ cat <filename source> >> <destination filename>

$ cat <source filename> >> <destination filename> it is avoid overwriting.

**Options : -**

-n content of file with numbers included with blank lines.

**Syntax :**

$cat –n <filename>

**4.SORTING A FILE :**

To sort the contents in alphabetical order in reverse order.

**Syntax :**

$sort <filename >

**Option :** $ sort –r <filename>

**5.COPYING CONTENTS FROM ONE FILE TO ANOTHER :**

To copy the contents from source to destination file . so that both contents are same.

**Syntax :**

$cp <source filename> <destination filename>

$cp <source filename path > <destination filename path>

**6.MOVE Command :**

To completely move the contents from source file to destination file and to remove the source file.

**Syntax :**

$ mv <source filename> <destination filename>

**7.REMOVE Command :**

To permanently remove the file we use this command .

**Syntax :**

$rm <filename>

**8.WORD Command :**

To list the content count of no of lines , words, characters .

**Syntax :**

$wc<filename>

**Options :**

-c – to display no of characters.

-l – to display only the lines.

-w – to display the no of words.

**9.LINE PRINTER :**

To print the line through the printer, we use lp command.

**Syntax :**

$lp <filename>

**10.PAGE Command :**

This command is used to display the contents of the file page wise & next page can be viewed by pressing the enter key.

**Syntax :**

$pg <filename>

**11. FILTERS AND PIPES**

**HEAD :** It is used to display the top ten lines of file.

**Syntax:** $head<filename>

**TAIL :** This command is used to display the last ten lines of file.

**Syntax:** $tail<filename>

**PAGE :** This command shows the page by page a screenfull of information is displayed after which the page command displays a prompt and passes for the user to strike the enter key to continue scrolling.

**Syntax:** $ls –a\p

**MORE :** It also displays the file page by page .To continue scrolling with more command , press the space bar key.

**Syntax:** $more<filename>

**GREP :**This command is used to search and print the specified patterns from the file. Sy**ntax:** $grep [option] pattern <filename>

**SORT :** This command is used to sort the datas in some order.

**Syntax:** $sort<filename>

**PIPE :** It is a mechanism by which the output of one command can be channeled into the input of another command.

**Syntax:** $who | wc-l

**TR :**The tr filter is used to translate one set of characters from the standard inputs to another.

**Syntax:** $tr “[a-z]” “[A-Z]”

**Vi editor**

The VI editor is a fast and powerful text editor of Unix system.

**SAVING AND QUITING FROM vi :-**

**1.<ESC> w Command :**

To save the given text present in the file.

**Syntax :** <ESC> : w

2.<ESC> q! Command :

To quit the given text without saving.

Syntax : <ESC> :q!

3.<ESC> wq Command :

This command quits the vi editor after saving the text in the mentioned file.

Syntax : <ESC> :wq

4.<ESC> x Command :

This command is same as „wq‟ command it saves and quit.

Syntax : <ESC> :x

5.<ESC> q Command :

This command would quit the window but it would ask for again to save the file.

Syntax : <ESC> : q