**Assignment No :- 1**

**Program Statement:-**

Write a shell script / program to print a reverse number of a given number.

**Program Algorithm:-**

**Description:**

This is a program to reverse a number. We take a number a reverse it by using ‘rev’ command, and display it.

**Steps:**

**Step 1** print “Enter a number”

input n

**Step 2** n=`echo $n|rev`

[ rev is a command that returns reverse of the hole inputted string. ]

**Step 3** print “Number in reverse order =” n

**Step 4** Exit from the program.

**Shell Script:-**

# Write a shell script / program to print a reverse number of a given number.

echo -n "Enter a Number ="

read n

n=`echo "$n"|rev`

echo "Number in reverse order =$n"

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 4.sh

Enter a Number =14

Number in reverse order =41

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 4.sh

Enter a Number =483

Number in reverse order =384

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 4.sh

Enter a Number =64064

Number in reverse order =46046

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* In this program we use rev command to execute it.
* The user defined variables are not needed to be initialized.

**Assignment No :- 2**

**Program Statement:-**

Write a shell script / program to determine whether the number is prime or not.

**Program Algorithm:-**

**Description:**

This is a program to check a number that is prime or not. Here we take a number and if it is a prime number then display prime otherwise not prime.

**Steps:**

**Step 1** print “Enter a Number =”

input n

flag 🡨 0

i 🡨 2

**Step 2** Loop continues when i less than or equal to n/2 true

If n mod I equal to 0 then

flag 🡨 1

goto **Step 3**

[ End of if ]

i 🡨 i+1

[ End of Loop ]

**Step 3** If n not equal to 1 AND flag equal to 0 then

print “n is a Prime Number.”

else

print “n is Not a Prime Number.”

[ End of if ]

**Step 4** Exit from the program.

**Shell Script:-**

# Write a shell script / program to determine whether the number is prime or not.

echo -n "Enter a Number ="

read n

flag=0

for ((i=2;i<=n/2;i++))

do

if [ $((n%i)) -eq 0 ]

then

flag=1

break

fi

done

if [ $flag -eq 0 -a $n -ne 1 ]

then

echo "$n is a Prime Number."

else

echo "$n is Not a Prime Number."

fi

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 5.sh

Enter a Number =23

23 is a Prime Number.

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 5.sh

Enter a Number =18

18 is Not a Prime Number.

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 5.sh

Enter a Number =103

103 is a Prime Number.

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 5.sh

Enter a Number =97

97 is a Prime Number.

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 5.sh

Enter a Number =91

91 is Not a Prime Number.

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 3**

**Program Statement:-**

Write a shell script / program to determine whether a particular user logged into the system or not.

**Program Algorithm:-**

**Description:**

This is a program to check that a user is logged in or not. Here we take a user name and if it is already logged in then prints it’s name otherwise we wait for that user when he/She login and then display the waiting time we were waited for that user.

**Steps:**

**Step 1** print “Enter the User Name =”

input u

t 🡨 0

f 🡨 0

**Step 2** Loop starts

ch 🡨 `who| grep -i $u`

[ here who command finds the all user that currently logged in and grep -i finds that string is present into that fie or not and if not exist then returns it otherwise return

nothing. ]

**Step 2.1** If ch not equal to nothing then

If f equal to 0 then

print u “is already logged in.”

else

print u “is logged in after ” t “minutes.”

[ End of if ]

exit from the program.

**Step 2.2** else

wait for 60 seconds

t 🡨 t+1

f 🡨 1

[ End of if ]

[ End of Loop ]

**Step 3** Exit from the program.

**Shell Script:-**

# Write a shell script / program to determine whether a particular user logged into the system or not.

echo -n "Enter the User Name ="

read u

t=0

f=0

while true

do

ch=`who|grep -i $u`

if [ "$ch" != "" ]

then

if [ $f -eq 0 ]

then

echo "$u is already logged in."

else

echo "$u is logged in after $t minutes."

fi

exit

else

sleep 60

t=$(($t+1))

f=1

fi

done

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 6.sh

Enter the User Name =susovan

susovan is already logged in.

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* In this program we use “who” and “grep” command to execute it.
* The user defined variables are not needed to be initialized.

**Assignment No :- 4**

**Program Statement:-**

Write a shell script /program that reads an integer and test whether it is divisible by 11 using divisibility rule.

**Program Algorithm:-**

**Description:**

This is a program to check that a number is divisible or not. Here we take a integer number and check it by 11’s divisibility rule that is divisible or not, if yes then print divisible otherwise print not divisible.

**Steps:**

**Step 1** print “Enter a Number =”

input n

i 🡨 1

s 🡨 0

**Step 2** Loop starts

ch=`echo “$n”|cut -c $i`

[ cut -c cut a character of the string from which location that passes into it and returns that character. ]

**Step 2.1** If ch equal to nothing then

goto the **Step 3**

[ end of if ]

**Step 2.2** If I mod 2 equal to 1 then

s 🡨 s+ch

else

s 🡨 s-ch

[ end of if ]

i 🡨 i+1

[ end of Loop ]

**Step 3** If s mod 11 equal to 0 then

print n “is Divisible by 11.”

else

print n “is Not Divisible by 11.”

[ end of If ]

**Step 4** Exit from the program.

**Shell Script:-**

# Write a shell script /program that reads an integer and test whether it is divisible by 11 using divisibility rule.

echo -n "Enter a Number ="

read n

i=1

s=0

while true

do

ch=`echo "$n"|cut -c $i`

if [ "$ch" = "" ]

then

break

fi

if [ $(($i%2)) -eq 1 ]

then

s=$(($s+$ch))

else

s=$(($s-$ch))

fi

i=$(($i+1))

done

if [ $(($s%11)) -eq 0 ]

then

echo "$n is Divisible by 11."

else

echo "$n is Not Divisible by 11."

fi

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 7.sh

Enter a Number =146

146 is Not Divisible by 11.

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 7.sh

Enter a Number =121

121 is Divisible by 11.

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 5.i**

**Program Statement:-**

Write a shell / program to print a pattern.

\*

\* \*

\* \* \*

\* \* \* \*

...

**Program Algorithm:-**

**Description:**

This is a program to print a pattern. Here we take the lines numbers as input and prints the corresponding pattern.

**Steps:**

**Step 1** print “Enter the Line Number =”

input n

i 🡨 0

**Step 2** Loop continue when i less than n true

j 🡨 0

**Step 2.1** Loop continue when j less than or equal to i true

print “\*”

j 🡨 j+1

[ end of Loop ]

print a new line.

i 🡨 i+1

[ End of Loop ]

**Step 3** Exit from the program.

**Shell Script:-**

# Write a shell / program to print the following pattern

# \*

# \* \*

# \* \* \*

# \* \* \* \*

# ...

echo -n "Enter the Line Number ="

read n

for ((i=0;i<n;i++))

do

for ((j=0;j<=i;j++))

do

echo -n "\* "

done

echo

done

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 8a.sh

Enter the Line Number =6

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

\* \* \* \* \* \*

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 5.ii**

**Program Statement:-**

Write a shell / program to print a pattern.

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

.........

**Program Algorithm:-**

**Description:**

This is a program to print a pattern. Here we take the lines numbers as input and prints the corresponding pattern.

**Steps:**

**Step 1** print “Enter the Line Number =”

input n

i 🡨 0

**Step 2** Loop continue when i less than n true

j 🡨 n-1

**Step 2.1** Loop continue when j greater than to i true

print blank space

j 🡨 j-1

[ end of Loop ]

**Step 2.2** j 🡨 0

Loop continue when j is less than to i\*2+1 true

print “\*”

j 🡨 j+1

[ end of Loop ]

print a new line.

i 🡨 i+1

[ End of Loop ]

**Step 3** Exit from the program.

**Shell Script:-**

# Write a shell / program to print the following pattern

# \*

# \* \* \*

# \* \* \* \* \*

# \* \* \* \* \* \* \*

# .........

echo -n "Enter the Line Number ="

read n

for ((i=0;i<n;i++))

do

for ((j=n-1;j>i;j--))

do

echo -n " "

done

for ((j=0;j<i\*2+1;j++))

do

echo -n "\* "

done

echo

done

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 8b.sh

Enter the Line Number =7

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \* \* \* \* \*

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 5.iii**

**Program Statement:-**

Write a shell / program to print a pattern.

1

0 1

1 0 1

0 1 0 1

.........

**Program Algorithm:-**

**Description:**

This is a program to print a pattern. Here we take the lines numbers as input and prints the corresponding pattern.

**Steps:**

**Step 1** print “Enter the Line Number =”

input n

i 🡨 0

str 🡨 “”

**Step 2** Loop continue when I less than n true

If i mod 2 equal to 0 then

str 🡨 “1 ”str

else

str 🡨 “0 ”str

[ end of If ]

print str with new line.

i 🡨 i+1

[ end of Loop ]

**Step 3** Exit from the program.

**Shell Script:-**

# Write a shell / program to print the following pattern

# 1

# 0 1

# 1 0 1

# 0 1 0 1

# .........

echo -n "Enter the Line Number ="

read n

str=""

for ((i=0;i<n;i++))

do

if [ $(($i%2)) -eq 0 ]

then

str="1 "$str

else

str="0 "$str

fi

echo "$str"

done

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 8c.sh

Enter the Line Number =8

1

0 1

1 0 1

0 1 0 1

1 0 1 0 1

0 1 0 1 0 1

1 0 1 0 1 0 1

0 1 0 1 0 1 0 1

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 5.iv**

**Program Statement:-**

Write a shell / program to print a pattern.

sinx = x - x^3/3! + x^5/5! - x^7/7! + ......

**Program Algorithm:-**

**Description:**

This is a program to print a pattern. Here we take the item numbers as input and prints the corresponding pattern.

**Steps:**

**Step 1** print “Enter the term numbers =”

input n

i 🡨 1

print “Sin x = x ”

**Step 2** Loop continue when i less than n true

If i mod 2 equal to 0 then

print “+ ” (2\*i+1)/(2\*i+1) “! ”

else

print “- ” (2\*i+1)/(2\*i+1) “! ”

[ end of If ]

i 🡨 i+1

[ End of Loop ]

**Step 3** Exit from the program.

**Shell Script:-**

# Write a shell / program to print the following pattern

# sinx = x - x^3/3! + x^5/5! - x^7/7! + ......

echo -n "Enter the term numbers ="

read n

echo -n "Sin x = x "

for ((i=1;i<n;i++))

do

if [ $(($i%2)) -eq 0 ]

then

echo -n "+ x^$((2\*$i+1))/$((2\*$i+1))! "

else

echo -n "- x^$((2\*$i+1))/$((2\*$i+1))! "

fi

done

echo

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 8d.sh

Enter the term numbers =10

Sin x = x - x^3/3! + x^5/5! - x^7/7! + x^9/9! - x^11/11! + x^13/13! - x^15/15! + x^17/17! - x^19/19!

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 6**

**Program Statement:-**

Write a shell script /program to generate a possible combinations of 1,2 and 3.

**Program Algorithm:-**

**Description:**

This is a program to print the all possible combination of 1,2 and 3. Here we use 3 Loop to print the all possible combination.

**Steps:**

**Step 1** print “All possible combinations of 1, 2 and 3 :-”

**Step 2** for i in (1,2,3)

**Step 3** for j in (1,2,3)

**Step 4** for k in (1,2,3)

print value of ijk

[ end of Loop ]

[ end of Loop ]

[ end of Loop ]

**Step 5** Exit from the program.

**Shell Script:-**

# Write a shell script /program to generate a possible combinations of 1,2 and 3.

echo "All possible combinations of 1, 2 and 3 :-"

for i in 1 2 3

do

for j in 1 2 3

do

for k in 1 2 3

do

echo "$i$j$k"

done

done

done

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 10.sh

All possible combinations of 1, 2 and 3 :-

111

112

113

121

122

123

131

132

133

211

212

213

221

222

223

231

232

233

311

312

313

321

322

323

331

332

333

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 7**

**Program Statement:-**

Write a menu driven shell script / program using switch statement.

**Program Algorithm:-**

**Description:**

This is a program to print a pattern.

**Steps:**

**Step 1** print “1. Add Numbers.\n2. Subtract Numbers.\n3. Multiply Numbers.\n4. Divide Numbers.\n5. Exit.”

**Step 2** print “Enter Your choice =”

input n

print “Enter A =”

input a

print “Enter B =”

input b

**Step 3** If n equal to 1 then

print “Ans of a+b =” a+b

goto **Step 9**

[End of if ]

**Step 4** If n equal to 2 then

print “Ans of a-b =” a-b

goto **Step 9**

[End of if ]

**Step 5** If n equal to 3 then

print “Ans of a\*b =” a\*b

goto **Step 9**

[End of if ]

**Step 6** If n equal to 4 then

print “Ans of a/b =” a/b

goto **Step 9**

[End of if ]

**Step 7** If n equal to 5 then

goto **Step 9**

[End of if ]

**Step 8** If n not equal to 1,2,3,4,5 then

print “Worng choice”

goto **Step 9**

[End of if ]

**Step 9** Exit from the program.

**Shell Script:-**

# Write a menu driven shell script / program using switch statement that has the following options :

# 1. Add numbers

# 2. Subtract numbers

# 3. Multiply numbers

# 4. Divide numbers

# 5. Exit

echo -e "1. Add Numbers.\n2. Subtract Numbers.\n3. Multiply Numbers.\n4. Divide Numbers.\n5. Exit."

echo -n "Enter your Choice ="

read n

echo -n "Enter A ="

read a

echo -n "Enter B ="

read b

case $n in

1) echo "Ans of $a+$b =`echo "scale=2;$a+$b"|bc`"

;;

2) echo "Ans of $a-$b =`echo "scale=2;$a-$b"|bc`"

;;

3) echo "Ans of $a\*$b =`echo "scale=2;$a\*$b"|bc`"

;;

4) echo "Ans of $a/$b =`echo "scale=2;$a/$b"|bc`"

;;

5) exit

;;

\*) echo "Wrong Choice."

;;

esac

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 11.sh

1. Add Numbers.

2. Subtract Numbers.

3. Multiply Numbers.

4. Divide Numbers.

5. Exit.

Enter your Choice =1

Enter A =10

Enter B =35

Ans of 10+35 =45

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 11.sh

1. Add Numbers.

2. Subtract Numbers.

3. Multiply Numbers.

4. Divide Numbers.

5. Exit.

Enter your Choice =2

Enter A =48

Enter B =129

Ans of 48-129 =-81

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 11.sh

1. Add Numbers.

2. Subtract Numbers.

3. Multiply Numbers.

4. Divide Numbers.

5. Exit.

Enter your Choice =3

Enter A =30

Enter B =5

Ans of 30\*5 =150

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 11.sh

1. Add Numbers.

2. Subtract Numbers.

3. Multiply Numbers.

4. Divide Numbers.

5. Exit.

Enter your Choice =9

Wrong Choice.

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 11.sh

1. Add Numbers.

2. Subtract Numbers.

3. Multiply Numbers.

4. Divide Numbers.

5. Exit.

Enter your Choice =4

Enter A =125

Enter B =6

Ans of 125/6 =20.83

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 8**

**Program Statement:-**

Write a shell script / program to count number of lines and words in a file.

**Program Algorithm:-**

**Description:**

This is a program to count number of lines and words in a file. Here take a file name as input and we find the line number and word number of that file.

**Steps:**

**Step 1** print “Enter the File name =”

input fn

ln 🡨 0

w 🡨 0

flag 🡨 0

**Step 2** open the file fn in output mode

**Step 3** Loop continue when read l true

[ read l means that it read every line from the file that opens in output mode. ]

i 🡨 1

**Step 3.1** Loop starts

ch =`echo “$l”|cut –c $i`

[ cut -c cut a character of the string from which location that passes into it and returns that character. ]

**Step 3.1.1** If ch equal to nothing then

goto **Step 3.2**

[ End of if ]

**Step 3.1.2** if ch equal to between one of them “ ”, “,”, “.”, “;”, “?” then

if flag equal to 0 then

w 🡨 w+1

flag 🡨 1

[ end of If ]

else

flag 🡨 0

[ End of if ]

i 🡨 i+1

[End of Loop]

**Step 3.2** if flag equal to 0 then

w 🡨 w+1

[ End of if ]

ln 🡨 ln +1

[ End Of Loop ]

**Step 4** print in file fn words =w and line =ln

**Step 5** Exit from the program.

**Shell Script:-**

# Write a shell script / program to count number of lines and words in a file.

echo -n "Enter the File name ="

read fn

ln=0

w=0

flag=0

exec<$fn

while read l

do

i=1

while true

do

ch=`echo "$l"|cut -c $i`

if [ "$ch" = "" ]

then

break

fi

if [ "$ch" = " " -o "$ch" = "," -o "$ch" = "." -o "$ch" = ";" -o "$ch" = "?" ]

then

if [ $flag -eq 0 ]

then

w=$(($w+1))

flag=1

fi

else

flag=0

fi

i=$(($i+1))

done

if [ $flag -eq 0 ]

then

w=$(($w+1))

fi

ln=$((ln+1))

done

echo "In file '$fn'"

echo "Words =$w"

echo "lines =$ln"

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 13.sh

Enter the File name =a.txt

In file 'a.txt'

Words =578

lines =184

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 13.sh

Enter the File name =13.sh

In file '13.sh'

Words =142

lines =40

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.
* In this program we use “cut” command to execute it.

**Assignment No :- 9**

**Program Statement:-**

Write a shell script / program to find the sum and average of first "N" numbers (20<=N=<50).

**Program Algorithm:-**

**Description:**

This is a program to find the sum and average of first "N" numbers (20<=N=<50).User will asked to input N. The program must check the validity of "N" as input on by user.

**Steps:**

**Step 1** print “Enter the N (20<=N<=50) =”

input n

**Step 2** if n is greater than 50 OR less than 20 then

print “Your data is not in range.”

goto **Step 6**

**Step 3** sum 🡨 0

i 🡨 1

**Step 4** Loop continue when i less than or equal to n true

sum 🡨 sim + i

i 🡨 i+1

[ end of Loop ]

**Step 5** print “Sum =” sum

print “Avarage =” sum/n

**Step 6** Exit from the program.

**Shell Script:-**

# Write a shell script / program to find the sum and average of first "N" numbers (20<=N=<50).User will asked to input N. The program must check the validity of "N" as input on by user.

echo -n "Enter the N (20<=N<=50) ="

read n

# Input condition checking

if [ $n -gt 50 -o $n -lt 20 ]

then

echo "Your Entered Number is Not in Range."

exit

fi

# Calculation

sum=0

for ((i=1;i<=$n;i++))

do

sum=$(($sum+$i))

done

echo "Sum =$sum"

echo "Average =`echo "scale=2;$sum/$n"|bc`"

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 14.sh

Enter the N (20<=N<=50) =25

Sum =325

Average =13.00

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 14.sh

Enter the N (20<=N<=50) =30

Sum =465

Average =15.50

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 14.sh

Enter the N (20<=N<=50) =60

Your Entered Number is Not in Range.

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 14.sh

Enter the N (20<=N<=50) =10

Your Entered Number is Not in Range.

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 10**

**Program Statement:-**

Write a shell script / program to print a text file in reverse way.

**Program Algorithm:-**

**Description:**

This is a program to print a text file in reverse way .Here we take a file name as a input and print that file in reverse way.

**Steps:**

**Step 1** print “Enter a File Name =”

input fn

str 🡨 “”

**Step 2** open the file fn in output mode

**Step 3** Loop continue when read l true

[ read l means that it read every line from the file that opens in output mode. ]

str 🡨 l“\n”str

[ End of Loop ]

**Step 4** print str

**Step 5** Exit from the program.

**Shell Script:-**

# Write a shell script / program to print a text file in reverse way.

echo -n "Enter the File name ="

read fn

str=""

exec<$fn

while read l

do

str=$l"\n"$str

done

echo -e "$str"

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ cat 1.sh

# Assignment No :- 1

# Write a shell script / program to generate Fibonacci series up to a certain number.

echo -n "Enter the Extreme Limit ="

read n

a=0

b=1

while [ $a -le $n ]

do

echo -n "$a "

c=$(($a+$b))

a=$b

b=$c

done

echo

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 15.sh

Enter the File name =1.sh

echo

done

b=$c

a=$b

c=$(($a+$b))

echo -n "$a "

do

while [ $a -le $n ]

b=1

a=0

read n

echo -n "Enter the Extreme Limit ="

# Write a shell script / program to generate Fibonacci series up to a certain number.

# Assignment No :- 1

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 11**

**Program Statement:-**

Write a shell script / program to find the sum of N numbers taken as parameter input.

**Program Algorithm:-**

**Description:**

This is a program to find the sum of N numbers taken as parameter input. Here we take the values form the terminal and find their Sum.

**Steps:**

**Step 1** If there are no argument passes then

print “No Argument Passes.”

goto **Step 5**

[ end of If ]

**Step 2** s 🡨 0

**Step 3** Loop starts for i in argument list

s 🡨 s+i

[ End of Loop ]

**Step 4** print “Sum of the Numbers =” s

**Step 5** Exit from the program.

**Shell Script:-**

# Write a shell script / program to find the sum of N numbers taken as parameter input.

if [ $# -eq 0 ]

then

echo "No Arguments Passes."

exit

fi

s=0

echo -n "Arguments Passes as the Parameter:- "

for i in $\*

do

echo -n "$i "

s=$(($s+$i))

done

echo -e "\nSum of that Numbers =$s"

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 16.sh

No Arguments Passes.

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 16.sh 1 2 3 4 5 6

Arguments Passes as the Parameter:- 1 2 3 4 5 6

Sum of that Numbers =21

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 16.sh 10 23 34 21 643 27

Arguments Passes as the Parameter:- 10 23 34 21 643 27

Sum of that Numbers =758

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.
* The inputs are take as parameter in the command line.

**Assignment No :- 12**

**Program Statement:-**

Write a shell script / program to rename a group of files.

**Program Algorithm:-**

**Description:**

This is a program to rename a group of files for example, rename all files where file name end with .HTM so that they end with .HTML.

**Steps:**

**Step 1** print “Enter the Existing file Extension ="

input efe

print “Enter the Converting file Extension ="

input cfe

**Step 2** for i in `ls \*efe`

[ ls command find the files and directories that in the current location now ls \*efe menas those files or directories that ended with efe content ]

ch 🡨 `echo “$i”| cut –d‘.’ –f 1`

[ cut –d‘.’ –f does the string divided into sub fields on basis of ‘.’ delimiter and returns the fields which we want. ]

ch 🡨 ch cfe

mv i ch

[ mv this command move the first file to the second file ]

[ end of Loop ]

**Step 3** print “Complete conversion.”

**Step 4** Exit from the program.

**Shell Script:-**

# Write a shell script / program to rename a group of files for example, rename all files where file name end with .HTM so that they end with .HTML.

echo -n "Enter the Existing file Extension ="

read efe

echo -n "Enter the Converting file Extension ="

read cfe

for i in `ls \*$efe`

do

ch=`echo "$i"|cut -d'.' -f 1`

ch=$ch$cfe

mv $i $ch

done

echo "Complete conversion"

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ ls

10.sh 12.sh 14.sh 16.sh 18.sh 19.sh 20.sh 22.sh 24.sh 28.sh 2.sh 31.sh 3.sh 5.sh 7.sh 8b.sh 8d.sh a.txt

11.sh 13.sh 15.sh 17.sh 18student.txt 1.sh 21.sh 23.sh 25.sh 27.sh 29.sh 30.sh 32.sh 4.sh 6.sh 8a.sh 8c.sh 9.sh

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 17.sh

Enter the Existing file Extension =.sh

Enter the Converting file Extension =.c

Complete conversion

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ ls

10.c 12.c 14.c 16.c 18.c 19.c 20.c 22.c 24.c 28.c 2.c 31.c 3.c 5.c 7.c 8b.c 8d.c a.txt

11.c 13.c 15.c 17.c 18student.txt 1.c 21.c 23.c 25.c 27.c 29.c 30.c 32.c 4.c 6.c 8a.c 8c.c 9.c

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.
* In this program we use “cut” and “ls” command to execute it.

**Assignment No :- 13**

**Program Statement:-**

Write a shell script to the division awarded to each student.

**Program Algorithm:-**

**Description:**

This is a program to the division awarded to each student. The students are awarded division as per following rules:- (1) Percentage above or equal to 60-1st division. (2) Percentage between 50 to 59 - 2nd division. (3) Percentage between 40 to 49 - 3rd division. (4) percentage lesa than 40 -FAIL.

**Steps:**

**Step 1** print “Data In Student File :-"

print the file 18student.txt

print "Students Divisions :-"

**Step 2** open the file 18student.txt in output mode

input l

[ read l means that it read every line from the file that opens in output mode. ]

**Step 3** Loop continue when read l true

[ read l means that it read every line from the file that opens in output mode. ]

set l

nm 🡨 $1

p 🡨 ($2+$3+$4+$5)/4

print nm

**Step 3.1** if p greater than or equal to 60 then

print “1st division”

[end of if]

**Step 3.2** else

if p greater than or equal to 50 then

print “2nd division”

**Step 3.3** else

if p greater than or equal to 40 then

print “3rd division”

**Step 3.4** else

print “Fail .”

[end of if]

[end of if]

[end of if]

**Step 4** Exit from the program.

**Shell Script:-**

# A student examination files containing the following format:

# Name sub1 sub2 sub3 sub4

# The students are awarded division as per following rules :

# i) Percentage above or equal to 60-1st division.

# ii) Percentage between 50 to 59 - 2nd division.

# iii) Percentage between 40 to 49 - 3rd division.

# iv) percentage lesa than 40 -FAIL.

# Write a shell script to the division awarded to each student.

echo "Data In Student File :-"

cat 18student.txt

echo

echo "Students Divisions :-"

exec<18student.txt

read l

while read l

do

set $l

nm=$1

p=`echo "scale=2;($2+$3+$4+$5)/4"|bc`

echo -n $nm" "

if [ 1 -eq `echo "$p>=60"|bc` ]

then

echo "1st division."

elif [ 1 -eq `echo "$p>=50"|bc` ]

then

echo "2nd division."

elif [ 1 -eq `echo "$p>=40"|bc` ]

then

echo "3rd division."

else

echo "Fail."

fi

done

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 18.sh

Data In Student File :-

NAME SUB\_1(Marks) SUB\_2(Marks) SUB\_3(Marks) SUB\_4(Marks)

Susovan 30 30 40 100

Shuvendu 30 100 40 30

Sourav 100 30 30 40

Sanada 80 80 80 80

Priya 80 80 80 80

Kanisha 80 80 80 80

Bikram 30 30 40 10

Students Divisions :-

Susovan 2nd division.

Shuvendu 2nd division.

Sourav 2nd division.

Sanada 1st division.

Priya 1st division.

Kanisha 1st division.

Bikram Fail.

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 14**

**Program Statement:-**

Write a shell script to find sort of n numbers of an array in ascending order.

**Program Algorithm:-**

**Description:**

This is a program to find sort of n numbers of an array in ascending order. Here we take total no of numbers and the numbers to sort them is ascending order.

**Steps:**

**Step 1** print “Enter the Size of Array =”

input n

print “Enter the Elements :-”

input a[]

[read all elements one by one]

print a[]

i 🡨 0

**Step 2** Loop continue when i less than n true

j 🡨 i+1

**Step 2.1** Loop continue when j less than n true

if a[i] greater than a[j] then

temp 🡨 a[i]

a[i] 🡨 a[j]

a[j] 🡨 temp

[ end of if ]

j 🡨 j+1

[ end of Loop ]

i 🡨 i+1

[ end of Loop ]

**Step 3** print “Elements After Sorting :-”

print a[]

**Step 4** Exit from the program.

**Shell Script:-**

# Write a shell script to find sort of n numbers of an array in ascending order.

echo -n "Enter the Size of Array ="

read n

echo "Enter the Elements :- "

for ((i=0;i<n;i++))

do

echo -n "A[$i] ="

read a[$i]

done

echo "Elements Are :-"

for ((i=0;i<n;i++))

do

echo -n "${a[$i]} "

done

for ((i=0;i<n;i++))

do

for ((j=i+1;j<n;j++))

do

if [ ${a[$i]} -gt ${a[$j]} ]

then

temp=${a[$i]}

a[$i]=${a[$j]}

a[$j]=$temp

fi

done

done

echo -e "\nElements After Sorting:-"

for ((i=0;i<n;i++))

do

echo -n "${a[$i]} "

done

echo

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 23.sh

Enter the Size of Array =4

Enter the Elements :-

A[0] =1

A[1] =9

A[2] =2

A[3] =6

Elements Are :-

1 9 2 6

Elements After Sorting:-

1 2 6 9

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 23.sh

Enter the Size of Array =9

Enter the Elements :-

A[0] =9

A[1] =3

A[2] =1

A[3] =5

A[4] =34

A[5] =0

A[6] =23

A[7] =45

A[8] =82

Elements Are :-

9 3 1 5 34 0 23 45 82

Elements After Sorting:-

0 1 3 5 9 23 34 45 82

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 23.sh

Enter the Size of Array =8

Enter the Elements :-

A[0] =8

A[1] =7

A[2] =6

A[3] =5

A[4] =4

A[5] =3

A[6] =2

A[7] =1

Elements Are :-

8 7 6 5 4 3 2 1

Elements After Sorting:-

1 2 3 4 5 6 7 8

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 15**

**Program Statement:-**

Write a shell script to find a factorial of a number using recursive process.

**Program Algorithm:-**

**Description:**

This is a program to find a factorial of a number using recursive process.

**Steps:**

**Step 1** fac() function

**Step 1.1** if n equal to 0 then

goto **Step 2**

[ end of if ]

**Step 1.2** f 🡨 f\*n

n 🡨 n-1

**Step 1.3** **call fac function**

**Step 2** print “Enter a Number =”

input n

f 🡨 1

print “Factorial of n is =”

**step 3** call fac

print f

**Step 4** Exit from the program.

**Shell Script:-**

# Write a shell script to find a factorial of a number using recursive process.

fac()

{

if [ $n -eq 0 ]

then

return

fi

f=$(($f\*$n))

n=$(($n-1))

fac

}

echo -n "Enter the Number ="

read n

f=1

echo -n "Factorial of $n is"

fac

echo -n " $f"

echo

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 27.sh

Enter the Number =4

Factorial of 4 is 24

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 27.sh

Enter the Number =10

Factorial of 10 is 3628800

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 27.sh

Enter the Number =6

Factorial of 6 is 720

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 27.sh

Enter the Number =12

Factorial of 12 is 479001600

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.

**Assignment No :- 16**

**Program Statement:-**

Write a shell script to find the no. of word that containing vowel in text file.

**Program Algorithm:-**

**Description:**

This is a program to find the no. of word that containing vowel in text file.

**Steps:**

**Step 1** print “Enter the File name =”

input fn

str 🡨 “”

**Step 2** open the file fn in output mode

**Step 3** Loop continue when read l true

[ read l means that it read every line from the file that opens in output mode. ]

**Step 3.1** for i in l

ch 🡨 `echo “$i”|grep –i [aeiou]`

[ here grep -i finds that string is present into that fie or not and if not exist then returns it otherwise return nothing. ]

if ch not equals to nothing then

str 🡨 ch“\n”str

[ end of if ]

[end of Loop]

[End of Loop]

**Step 4** ch 🡨 `echo –e “$str”|wc -l `

[ wc –l returns the total line number of that file passes throw the parameter ]

**Step 5** ch 🡨 `echo "$ch"|cut -d' ' -f 1`

[ cut –d‘.’ –f does the string divided into sub fields on basis of ‘ ’ delimiter and returns the fields which we want. ]

print “The number of words that have vowels is ” ch

**Step 6** Exit from the program.

**Shell Script:-**

# Write a shell script to find the no. of word that containing vowel in text file.

echo -n "Enter the File Name ="

read fn

str=””

exec<$fn

while read l

do

for i in $l

do

ch=`echo "$i"|grep -i [aeiou]`

if [ "$ch" != "" ]

then

str=$ch”\n”$str

fi

done

done

ch=`echo –e “$str”|wc -l `

ch=`echo "$ch"|cut -d' ' -f 1`

echo "In File '$fn' the number of words that containing vowels is $ch"

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 31.sh

Enter the File Name =4.sh

In File '4.sh' the number of words that containing vowels is 28

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 31.sh

Enter the File Name =5.sh

In File '5.sh' the number of words that containing vowels is 51

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 31.sh

Enter the File Name =a.txt

In File 'a.txt' the number of words that containing vowels is 756

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 31.sh

Enter the File Name =6.sh

In File '6.sh' the number of words that containing vowels is 56

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 31.sh

Enter the File Name =17.sh

In File '17.sh' the number of words that containing vowels is 54

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.
* In this program we use “cut” , “grep” , “wc” command to execute it.

**Assignment No :- 17**

**Program Statement:-**

Write a shell script to find a large name to its shorted representation.

**Program Algorithm:-**

**Description:**

This is a program to find a large name to its shorted representation.

**Steps:**

**Step 1** print “Enter a Name =”

input n

l 🡨 `echo “$n” | wc -w`

[ wc –w returns the word count from the file that passes throw by parameter. ]

l 🡨 numeric vlue of l

str 🡨 “”

i 🡨 1

**Step 2** Loop continue when i less than l true

ch 🡨 `echo “$n”|cut –d‘ ’ –f i`

[ cut –d‘.’ –f does the string divided into sub fields on basis of ‘ ’ delimiter and returns the fields which we want. ]

ch 🡨 `echo "$ch"|cut -c 1`

[ cut -c cut a character of the string from which location that passes into it and returns that character. ]

str 🡨 str ch “.”

i 🡨 i+1

[ end of Loop ]

**Step 3** ch 🡨 `echo $n|cut -d' ' -f $i`

[ cut –d‘.’ –f does the string divided into sub fields on basis of ‘ ’ delimiter and returns the fields which we want. ]

str 🡨 str ch

print “Short from of the name = ” str

**Step 4** Exit from the program.

**Shell Script:-**

# Write a shell script to find a large name to its shorted representation.

echo -n "Enter a Name ="

read n

l=`echo "$n"|wc -w`

l=`expr $l`

str=""

for ((i=1;i<l;i++))

do

ch=`echo $n|cut -d' ' -f $i`

ch=`echo "$ch"|cut -c 1`

str=$str$ch"."

done

ch=`echo $n|cut -d' ' -f $i`

str=$str$ch

echo "Short from of the name =$str"

**Output:-**

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 32.sh

Enter a Name =Susovan Das

Short from of the name =S.Das

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 32.sh

Enter a Name =Sapen Kumar Dutta

Short from of the name =S.K.Dutta

susovan@susovan-Inspiron-3542:~/sh\_programs/Assignment$ sh 32.sh

Enter a Name =Aritra Roy

Short from of the name =A.Roy

**Discussion:-**

* Proper space is maintained between the brackets to ensure smooth execution of the program.
* The program is executed with “bash” shell to ensure proper execution.
* The user defined variables are not needed to be initialized.
* In this program we use “cut” command to execute it.