**Ex.No: 2 SHELL PROGRAMMING**

**Date : 04.03.2021**

**Aim**

To write the following shell programs and execute in unix environment.

**1.Write a shell program for getting and displaying academic details. Inputs are name, rollno and 3 subject marks. Output is To display Name, rollno, 3 subject marks, total marks, and average marks.**

**Algorithm:**

**1.** Start

**2.** Create a file using vi command with filename.sh

**3.** Using ‘read’ command, read name,rollno,mark1,mark2,mark3 from the user.

**4.** Using ‘expr’ command, evaluate the total marks by adding mark1,mark2,mark3 store it in tot variable) and average marks by dividing total marks by total no of marks i.e., 3 (store it in avg variable)

**5.** Using ‘echo’ command, print the student’s name,rollno,total marks,average to the user.

**6.** Stop.

**Program:**

echo "Enter name:"

read name

echo "Enter Rollno:"

read rno

echo "Enter 3 subject Marks:"

read m1

read m2

read m3

tot=`expr $m1 + $m2 + $m3`

avg=`expr $tot / 3`

echo "Academic details:

Name: $name

Roll no: $rno

Marks 1: $m1

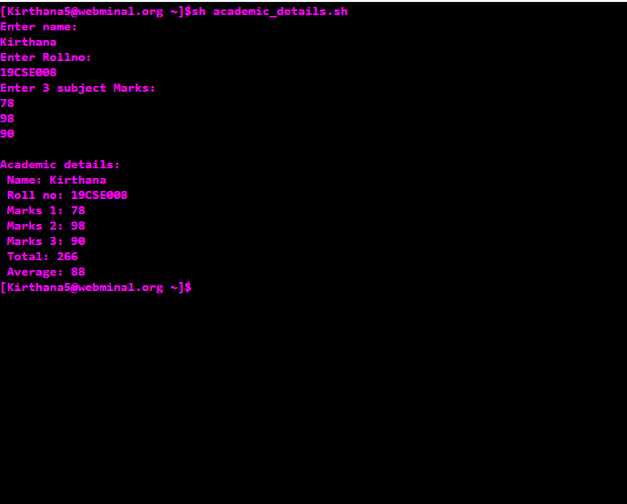
Marks 2: $m2

Marks 3: $m3

Total: $tot

Average: $avg"

**Sample Output:**



**2.Write a shell program to do arithmetic operations.**

**Algorithm:**

**1.** Start

**2.** Create a file using vi command with filename.sh

**3.** Using ‘echo’ command, print the statement “Enter the value of two numbers:”

**4**. Using ‘read’ command, read the value of a and b from the user.

**5.** Using ‘expr’ command, evaluate arithmetic operations such as add,subtract,multiply, divide,modulo for the values of a and b.

**6.** Using ‘echo’ command, print the addition,subtraction,multiplication,division and modulo result to the user.

**7.** Stop.

**Program:**

echo "Enter two numbers:"

read a

read b

c=`expr $a + $b`

echo "Addition result: $c"

c=`expr $a - $b`

echo "Subtraction result: $c"

c=`expr $a \\* $b`

echo "Multiplication result: $c"

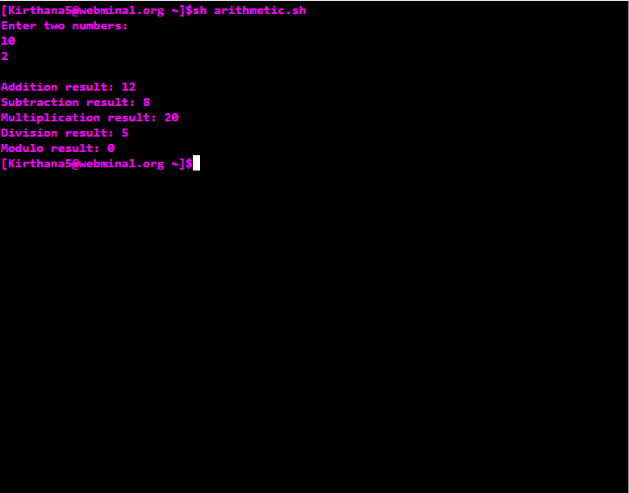
c=`expr $a / $b`

echo "Division result: $c"

c=`expr $a % $b`

echo "Modulo result: $c"

**Sample Output:**



**3.Write a shell program to check for positive number, negative number and zero.**

**Algorithm:**

**1.** Start

**2.** Create a file using vi command with filename.sh

**3.** Using ‘echo’ command, print the statement “Enter a number:”

**4**. Using ‘read’ command, read the value of a from the user.

**5.** Check whether the number is positive ,negative or zero by using ‘if’ conditonal statements

**6.** After ‘if’ give the condition as $a -gt 0 and next give the keyword ‘then’ and if the condition satisfies ,print the statement as “a is a positive number”

**7.** If the given condition fails,give the condition as $a -lt 0 in ‘elif’ statement and next give the keyword ‘then’ and if the condition satisfies ,print the statement as “a is a negative number”

**8**. If the none of the condition satisfies,print the statement “a is ZERO” in else statement and give ‘fi’ to terminate the if statements.

**9.** Stop.

**Program:**

echo "Enter an integer:"

read a

if [ $a -gt 0 ]

then

echo "$a is a Positive number !! "

elif [ $a -lt 0 ]

then

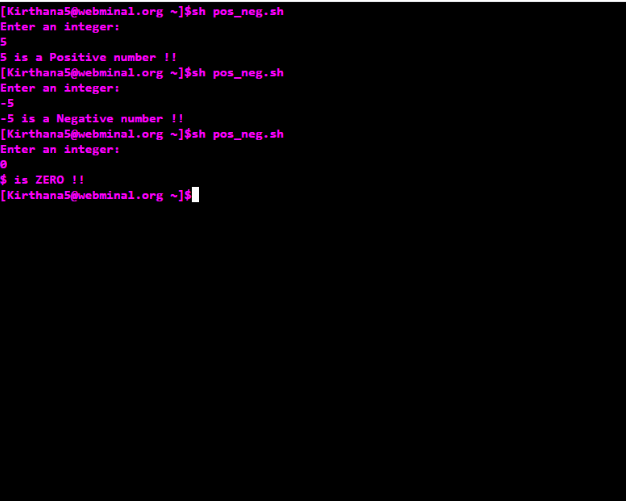
echo "$a is a Negative number !! "

else

echo "$a is ZERO !! "

fi

**Sample Output:**



**4.Write a shell program to print the combinations of 1,2 and 3.**

**Algorithm:**

**1**. Start

**2.** Create a file using vi command with filename.sh

**3.** Using 3 ‘for’ loops , we can print the combination of 123.

**4.** First give ‘for var in 1 2 3’ and then give ‘do’ keyword ,like wise create 3 for loops inside one another with 3 different variables.

**5.** Inside the third loop print the value of three variables.

**6.** Combination of 123 gets printed.

**7.** Using ‘done’ keyword, end the each for loops one after another.

**8.** Stop.

**Program:**

echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Combinations of 1 2 3\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

a=1

b=1

c=1

for a in 1 2 3

do

for b in 1 2 3

do

for c in 1 2 3

do

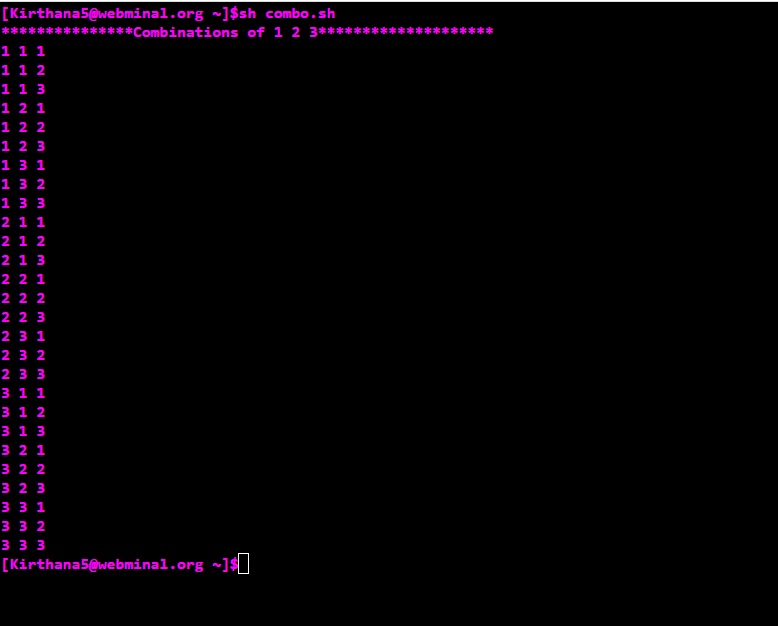
echo "$a $b $c"

done

done

done

**Sample Output:**

****

**5.Write a shell program to find and display the area of Rectangle, Square, Circle and Triangle using ‘case’ stmt.**

**Algorithm:**

**1.** Start

**2.** Create a file using vi command with filename.sh

**3.** Using ‘echo’ command, print the statement "1.Rectangle 2.Square 3.Circle 4.Triangle

Enter Ur Choice:"

**4**. Using ‘read’ command, read the value of a from the user

**5.** Inside case, give 4 cases for triangle, circle, square, rectangle respectively. User can choose any case to execute it.

**6.** In first case read the values of length and breadth from user and calculate and print the area of rectangle.

**7.** In second case ,read the values of side from user and calculate and print the area of square.

**8.** In third case, read the values of radius from user and calculate and print the area of circle.

**9.** In fourth case, read the values of base and height from user and calculate and print the area of triangle.

**10.** In default case, print "Enter valid choice :("

**11.** Terminate case with keyword ‘esac’

**12.** Stop

**Program:**

echo "1.Rectangle

2.Square

3.Circle

4.Triangle

Enter Ur Choice:"

read a

read area

case $a in

1) echo "Rectangle:

Enter length and breadth:"

read l

read b

area=`expr $l \\* $b`

echo "Area of rectangle: $area"

;;

2) echo "Square:

Enter the side:"

read aa

area=`expr $aa \\* $aa`

echo "Area of square: $area"

;;

3) echo "Circle:

Enter radius:"

read rr

area=`expr $rr \\* $rr \\* 22 / 7`

echo "Area of circle: $area"

;;

4) echo "Triangle:

Enter base and height:"

read b

read h

area=`expr $b \\* $h / 2`

echo "Area of triangle: $area"

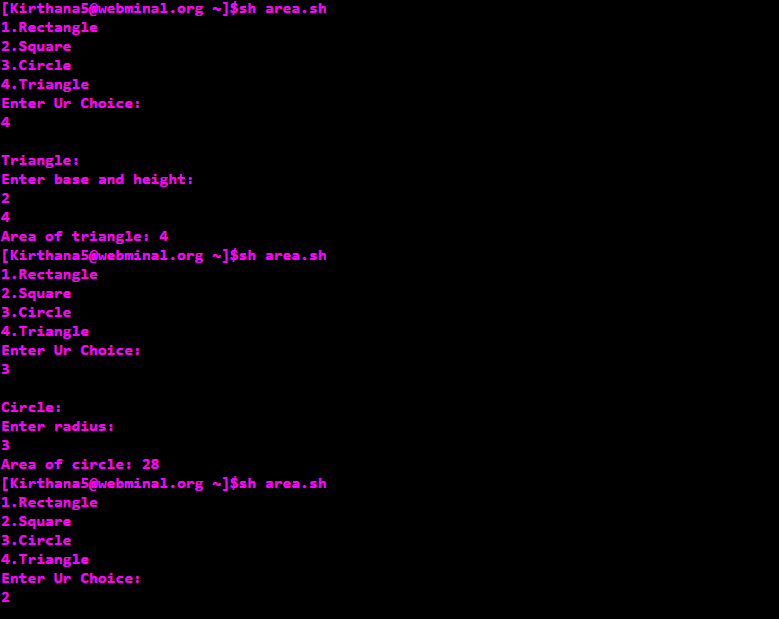
;;

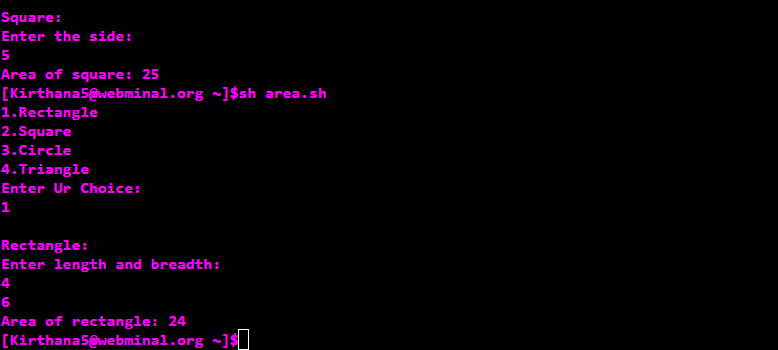
\*) echo "Enter valid choice :("

;;

esac

**Sample Output:**

****

****

**6.Write a shell program to concatenate two strings and find the length of the resultant string.**

**Algorithm:**

**1.** Start

**2.** Create a new file using vi command filename.sh

**3.** Get the two strings s1,s2 from the user using ‘read’ command.

**4.** Concatenate two strings by simply joining them and store in in the variable s3.

**5.** Print the concatenated string using ‘echo’ command.

**6**. Print the length of the concatenated string using the statement ${#s3}.

**7**. Stop.

**Program:**

echo "Enter the 2 strings:"

read s1

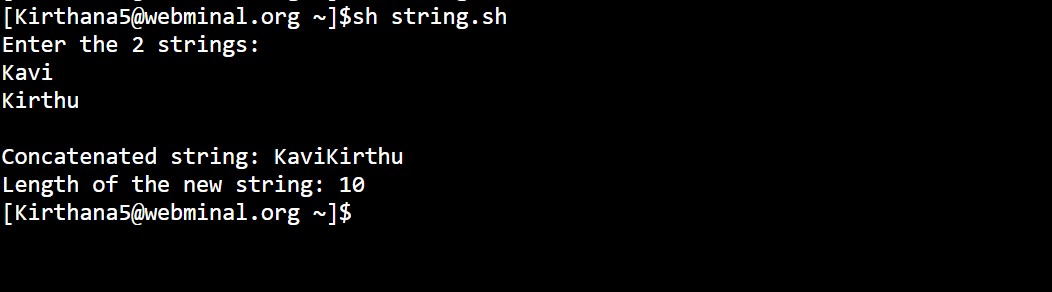
read s2

s3=$s1$s2

echo "Concatenated string: $s3 "

echo "Length of the new string: ${#s3}"

**Sample Output:**

****

**7.Write a shell program to display digits which are in Odd position in a given number.**

**Algorithm:**

**1**. Start

**2**. Create a new file using vi command filename.sh

**3**. Get the input number num from the user using ‘read’ command.

**4**. Declare on more variable a with value as 1.

**5**. Using ‘until’ loop with condition as ‘until value of a is greater than value of num’ do the step 6 and 7.

**6**. Print the digit in the ath column digit using ‘echo’ command.

**7**. Increment the value of a by 2.

**8**. Repeat step 6 and 7 until the condition fails.

**9**. Stop.

**Program:**

echo "Enter the number:"

read num

read a

a=1

until [ $a -gt ${#num} ]

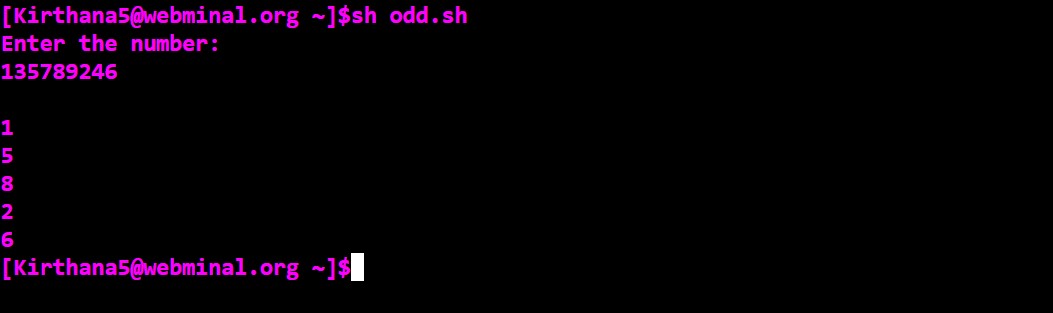
do

echo $num | cut -c $a

a=`expr $a + 2`

done

**Sample Output:**

****

**8.Write a shell program to search an element from the given array.**

**Algorithm:**

**1**. Start

**2**. Create a new file using vi command filename.sh

**3**. Declare an array a and get the size of array n from the user using ‘read’ command.

**4**. Declare a flag available with value as 0.

**5**. Get the array elements from the user using ‘for’ loop.

**6**. Display the array using the simple statement ${a[@]}.

**7**. Get the key element to be searched from the user using ‘read’ command.

**8**. In another ‘for’ loop(which runs up to the length of the array), search for the key in the array using ‘if’ statement.

**9**. In ‘if’ statement, specify the equality condition as ‘if the element at ith position matches the key’, set value of flag as 1.

**10**. Repeat the step 9 until the loop terminates.

**11**. Then using ‘if’ statement, check whether the value of flag matches 1 .

**12**. If it is true, print “Searched element is found” else print “No such element is found”.

**13**. Stop.

**Program:**

declare -a a

echo "Enter the size of array:"

read n

flag=0

echo "Enter the array elements:"

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

do

read ele

a[$i]=$ele

done

echo "The array: ${a[@]}"

echo "Enter the element to be searched:"

read key

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

do

if [ ${a[i]} -eq $key ]

then

flag=1

fi

done

if [ $flag -eq 1 ]

then

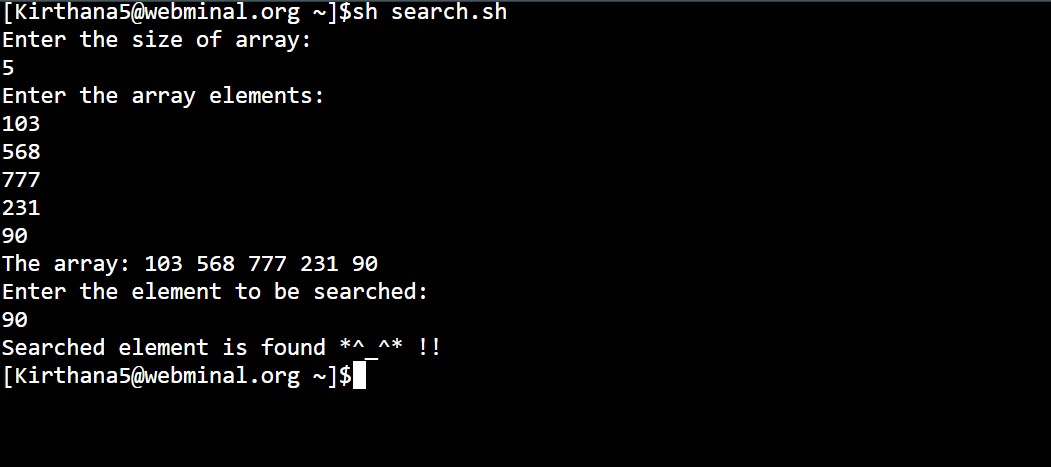
echo "Searched element is found \*^\_^\* !!"

else

echo "Ohooh!! No such element is found -\_-"

fi

**Sample Output:**

****

**9.Write a shell program to delete all zero sized files using ‘if’ and ‘for’.**

**Algorithm:**

**1**. Start

**2**. Create a new file using vi command filename.sh

**3**. Use ‘for’ loop which iterate over all the files in the root directory, and do the following steps.

**4**. Using ‘if’ statement, check whether the file has size using –s command.

**5**. If the file has size greater than zero, print that the “File has size > 0”

**6**. Else print “File is removed as its size is 0” and remove(delete) that file using rm command.

**7**. Repeat the steps 4 and 5 until the loop terminates.

**8**. Stop.

**Program:**

echo "Deleting all ZERO SIZED files"

for a in \*

do

if [ -s $a ]

then

echo "$a has a file size > 0"

else

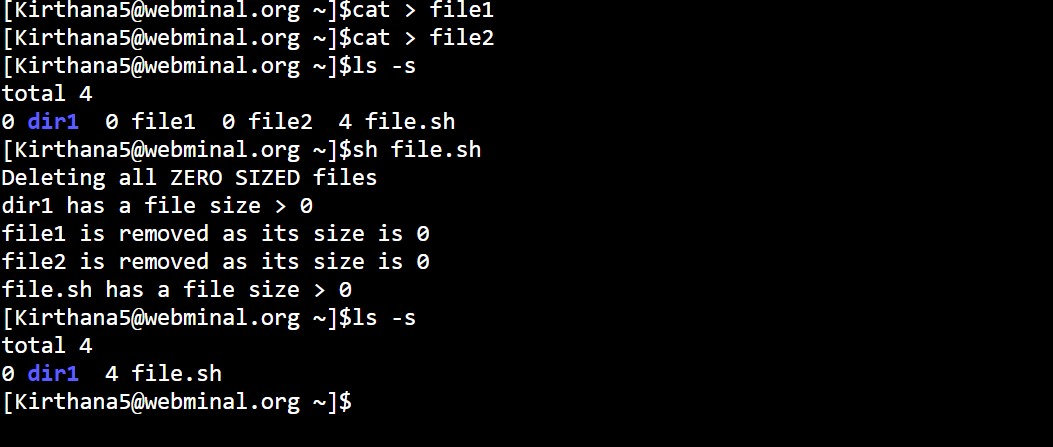
echo "$a is removed as its size is 0"

rm $a

fi

done

**Sample Output:**

****

**10.Write a shell program to reverse a number.**

**Algorithm:**

**1**. Start

**2**. Create a new file using vi command filename.sh

**3**. Get the input number n from the user using ‘read’ command.

**4**. Declare two more variables rem and rev and assign values as 0.

**5**. Create a temporary variable temp and store the value of n in it.

**6**. Using ‘while’ loop with condition ‘while value of temp is greater than 0’ do the following steps.

**7**. Using ‘expr’ command, evaluate the modulo of temp and 10 and store it in rem.

**8**. Using ‘expr’ command, evaluate the multiplication of rev and 10 and add rem with it and store it back in rem.

**9**. Using ‘expr’ command, evaluate the division of temp and 10 and store it in temp.

**10**. Repeat the steps 7,8 and 9 until the condition is satisfied.

**11**. Stop.

**Program:**

echo "Enter a number:"

read n

read temp

read rem

read rev

rem=0

rev=0

temp=$n

while [ $temp -gt 0 ]

do

rem=`expr $temp % 10`

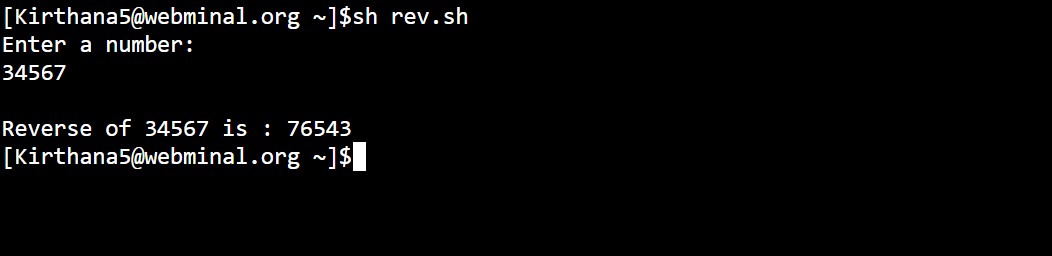
rev=`expr $rev \\* 10 + $rem`

temp=`expr $temp / 10`

done

echo "Reverse of $n is : $rev"

**Sample Output:**

****

|  |  |
| --- | --- |
| **Observation(20)** |  |
| **Record(5)** |  |
| **Total(25)** |  |
| **Initial** |  |

**Result:**

Thus the above shell programs are executed successfully.