CS623PE: SCRIPTING LANGUAGES LAB (Professional Elective - III)

III Year B.Tech. CSE II-Sem **L T P C**

0 0 2 1

Prerequisites: Any High-level programming language (C, C++)

Course Objectives:
 To Understand the concepts of scripting languages for developing web based projects
 To understand the applications the of Ruby, TCL, Perl scripting languages

Course Outcomes:
 Ability to understand the differences between Scripting languages and programming languages
 Able to gain some fluency programming in Ruby, Perl, TCL

List of Experiments

1. Write a Ruby script to create a new string which is n copies of a given string where n is a non-negative integer
2. Write a Ruby script which accept the radius of a circle from the user and compute the parameter and area.
3. Write a Ruby script which accept the user's first and last name and print them in reverse order with a space between them
4. Write a Ruby script to accept a filename from the user print the extension of that
5. Write a Ruby script to find the greatest of three numbers
6. Write a Ruby script to print odd numbers from 10 to 1
7. Write a Ruby scirpt to check two integers and return true if one of them is 20 otherwise return their sum
8. Write a Ruby script to check two temperatures and return true if one is less than 0 and the other is greater than 100
9. Write a Ruby script to print the elements of a given array
10. Write a Ruby program to retrieve the total marks where subject name and marks of a student stored in a hash
11. Write a TCL script to find the factorial of a number
12. Write a TCL script that multiplies the numbers from 1 to 10
13. Write a TCL script for Sorting a list using a comparison function
14. Write a TCL script to (i)create a list (ii)append elements to the list (iii)Traverse the list (iv)Concatenate the list
15. Write a TCL script to comparing the file modified times.
16. Write a TCL script to Copy a file and translate to native format.
17. a) Write a Perl script to find the largest number among three numbers.
b) Write a Perl script to print the multiplication tables from 1-10 using subroutines.
18. Write a Perl program to implement the following list of manipulating functions
a) Shift
b) Unshift
c) Push
19. a) Write a Perl script to substitute a word, with another word in a string.
b) Write a Perl script to validate IP address and email address.
20. Write a Perl script to print the file in reverse order using command line arguments

1. Write a Ruby script to create a new string which is n copies of a given string where n is a non-negative integer ?

Program :

```
puts"Enter a String Value:"
a = gets.chomp.to_s

puts"Enter an Integer Value:"
b=gets.chomp.to_i
        if b <0
                puts "Negative Number Cannot be Multiply"
        else
                puts (a*b)
        end
```

Output 1:

```
Enter a String Value:
Krupa sagar
Enter an Integer Value:
3
Krupa sagarKrupa sagarKrupa sagar
```

Output 2:

```
Enter a String Value:
Krupa sagar
Enter an Integer Value:
-3
Negative Number Cannot be Multiply
```

2. Write a Ruby script which accept the radius of a circle from the user and compute the parameter and area.

Program:

```
puts"Enter Radius of the Circcle:"
Radius = gets.chomp.to_i

Parameter = 2*Math::PI*Radius

Area = Math::PI*Radius*Radius

puts "parameter of Circle is:#{Parameter}"
puts "Area of the Circle is: #{Area}"
```

Output:

```
Enter Radius of the Circcle:
7
parameter of Circle is:43.982297150257104
Area of the Circle is: 153.93804002589985
```

3. Write a Ruby script which accept the user's first and last name and print them in reverse order

Program:

```
puts"Enter the First Name :"
f= gets.chomp.to_s
puts"Enter the Last Name :"
l=gets.chomp.to_s
Full_Name=f+"  "+l
puts"Full_Name:" +Full_Name.to_s
puts"Print First Name & Last Name in Reverse Order with Word wise :#{l} #{f}"
puts"Print First Name & Last Name in Reverse order with Character wise :
#{Full_Name.reverse}"
```

Output:

```
Enter the First Name :
Krupa
Enter the Last Name :
Sagar
Full_Name:Krupa   Sagar
Print First Name & Last Name in Reverse Order with Word wise :Sagar Krupa
Print First Name & Last Name in Reverse order with Character wise : ragaS   apurK
```

4. Write a Ruby script to accept a filename from the user print the extension of that

Program:

```
puts"Enter The File Name is :"
File_Name= gets.chomp.to_s

Extension_of_File_Name=File_Name.split(".")
puts "Extension of the File Name is: #{Extension_of_File_Name[1]}"


#puts File.extname ("sagar.txt")
#puts File.basename ("/home/system55/Documents/pks/r4.rb")
#puts File.dirname("/home/system55/Documents/pks/r4.rb")
#puts File.directory?("cats")
```

Output:

```
Enter The File Name is :
sagar.txt
Extension of the File Name is: txt
#.txt
#r4.rb
#/home/system55/Documents/pks
#false
```

5. Write a Ruby script to find the greatest of three numbers?

Program:

```
puts"Enter First Number a="
a=gets.chomp
puts"Enter Second Number b="
b=gets.chomp

puts"Enter Third Number c="
c=gets.chomp

if a>b && a>c
        puts "#{a} is greatest"

elsif b>a && b>c
        puts "#{b} is greatest"

elseif c>a && c>b
        puts "#{c} is greatest"
else
        puts"All Are Equal"
end
```

Output 1:

```
Enter First Number a=
25.12
Enter Second Number b=
34.2
Enter Third Number c=
2
34.2 is greatest
```

Output 2:

```
Enter First Number a=
10
Enter Second Number b=
10
Enter Third Number c=
10
All Are Equal
```

6. Write a Ruby script to print odd numbers from 10 to 1

Program:

```
puts "Enter an integer 10 :"
i=gets.chomp.to_i
puts "print odd numbers from 10 to 1 :"
while i>=0
        if(i%2!=0)
                puts i
        end
                i-=1
        end
```

Output:

```
Enter an integer 10 :
10
print odd numbers from 10 to 1 :
9
7
5
3
1
```

7. Write a Ruby scirpt to check two integers and return true if one of them is 20 otherwise return their sum

Program:

```
puts "Enter First Integer Value"
a=gets.chomp.to_i

puts"Enter Second Integer Value"
b=gets.chomp.to_i

if a==20 or b==20
        puts"The Value is : True"
else
        puts"The Value is :#{(a+b)}"
end
```

Output 1:

```
Enter First Integer Value
12
Enter Second Integer Value
13
The Value is :25
```

Output 2:

```
Enter First Integer Value
20
Enter Second Integer Value
12
The Value is : True
```

Output 3:

```
Enter First Integer Value
20
Enter Second Integer Value
20
The Value is : True
```

8. Write a Ruby script to check two temperatures and return true if one is less than 0 and the other is greater than 100

Program:

```
puts"Enter First Temperature"
a=gets.chomp.to_i

puts"Enter Second Temperature"
b=gets.chomp.to_i

if a<0 and b>100
        puts "The value is : True"
elsif b<0 and a>100
        puts "The value is : True"
else
        puts"The value is : False"
end
```

Output 1:

```
Enter First Temperature
-2
Enter Second Temperature
105
The value is : True
```

Output 2:

```
Enter First Temperature
105
Enter Second Temperature
-6
The value is : True
```

Output 3:

```
Enter First Temperature
56
Enter Second Temperature
90
The value is : False
```

9. Write a Ruby script to print the elements of a given array

Program:

```
puts ("Enter the size of an Array:")
n=Integer (gets.chomp)
a=Array.new(n)
puts("Enter the elements of Array :")
        for i in 0...n
                a[i]=gets
        end

puts ("Elements of the Array :")
        for i in 0...a.length do
                puts a[i]
        end
```

Output:

```
Enter the size of an Array:
5
Enter the elements of Array :
1
2.35
sagar
1+4i
789.3424634546576
Elements of the Array :
1
2.35
sagar
1+4i
789.3424634546576
```

10. Write a Ruby program to retrieve the total marks where subject name and marks of a student stored in a hash

Program:

```
student_marks={"Maths"=>89,"Science"=>78,"Sllab"=>75}
total_marks = 0
student_marks.each{|key,value|
        total_marks+=value}
puts "Total marks : #{+total_marks.to_s}"
```

Output:

```
Total marks : 242
```

11. Write a TCL script to find the factorial of a number?

Program:

```
#!/usr/bin/tclsh
puts "hello!"
proc factorial {n} {
if ($n==1) {
return 1
} else {
return [expr $n*[factorial [expr $n-1 ]]]
}
}
puts [factorial 9]
```

Output:

```
hello!
362880
```

12. Write a TCL script that multiplies the numbers from 1 to 10 ?

Program:

```
#!/usr/bin/tclsh
puts "Enter a number"
gets stdin x
puts "table"
for {set i 1} {$i<=10} {incr i} {
set mul [expr $x*$i]
puts $x*$i=$mul
}
```

Output:

```
Enter a number
7 table
7*1=7
7*2=14
7*3=21
7*4=28
7*5=35
7*6=42
7*7=49
7*8=56
7*9=63
7*10=70
```

13. Write a TCL script for Sorting a list using a comparison function

Program:

```
#!/usr/bin/tclsh
set var {8 6 5}
puts $var
puts "after sorting"
set var1 [lsort $var]
puts $var1
```

Output:

```
8 6 5
after sorting
5 6 8
```

14. Write a TCL script to (i)create a list (ii)append elements to the list (iii)Traverse the list (iv)Concatenate the list

Program 1:

```
#!/usr/bin/tclsh
set myVaribles {1 2 3}
puts $myVaribles
lappend myVaribles 4 5
puts $myVaribles
puts "Traverse"
foreach i $myVaribles {
puts $i
}
puts "concatinating list:"
set myVaribles2 { 8 9 }
set res [concat $myVaribles $myVaribles2]
puts $res
```

Output:

```
1 2 3
1 2 3 4 5
Traverse
1
2
3
4
5
concatinating list:
1 2 3 4 5 8 9
```

Program 2:

```
set L1 {1 2 3 }
puts $L1

lappend L1 4 5
puts "After append $L1"

puts "Traversing list"
set i 0
set len [llength $L1]
while {$i<$len} {
puts [lindex $L1 $i]
incr i
}

set L2 {-1 0}
puts "List 2 $L2"

set L3 [concat $L2 $L1]
puts "After concat $L3"
```

Output:

```
1 2 3
After append 1 2 3 4 5
Traversing list
1
2
3
4
5
List 2 -1 0
After concat -1 0 1 2 3 4 5
```

15. Write a TCL script to comparing the file modified times.

Program:

```
set tclfiles [glob *.tcl]
puts "Name - date of last modification:"
foreach f $tclfiles {
puts "$f- [clock format [file mtime $f] -format %x]";
puts "$f- [clock format [file atime $f] -format %x]";
}
```

Output:

```
Name - date of last modification:
com.tcl- 08/06/2021
com.tcl- 08/06/2021
list.tcl- 08/03/2021
list.tcl- 08/03/2021
mul.tcl- 08/03/2021
mul.tcl- 08/06/2021
file1.tcl- 08/04/2021
file1.tcl- 08/06/2021
native.tcl- 08/06/2021
native.tcl- 08/06/2021
tcl.tcl- 07/31/2021
tcl.tcl- 08/03/2021
file.tcl- 08/03/2021
file.tcl- 08/06/2021
sort.tcl- 08/03/2021
sort.tcl- 08/03/2021
hello.tcl- 08/06/2021
hello.tcl- 08/06/2021
```

16. Write a TCL script to Copy a file and translate to native format.

Program:

```
proc f_Copy {src dest} {
if [file isdirectory $src] {
file mkdir $dest
foreach f [global [file join $src *]] {
f_Copy $f[file join $dest [file tail $f]]
}
return
}
if [file isdirectory $dest] {
set dest [file join $dest [file tail $src]]
}
set in [open $src r]
set out [open $dest w+]
puts $out [read $in]
close $out;
close $in;
}
puts [f_Copy hello.tcl native.tcl]
```

Input:

```
puts "hello"
puts "enter a num"
gets stdin a
```

Output:

```
puts "hello"
puts "enter a num"
gets stdin a
```

17. a) Write a Perl script to find the largest number among three numbers.

Program:

```
#!/usr/bin/perl
print "enter a value:";
$a=<stdin>;
print "enter b value:";
$b=<stdin>;
print "enter c value:";
$b=<stdin>;
if($a > $b)
{
if($a > $b)
{
print "$a is the largest";
}
else
{
print "$a is the largest";
}
}
```

```
elsif($b > $c)
{
print "$b is the largest";
}
else
{
print "$c is the largest";
}
```

Output:

```
enter a value:1
enter b value:3
enter c value:5
5 is the largest
```

b) Write a Perl script to print the multiplication tables from 1-10 using subroutines.

Program:

```
#!/usr/bin/perl
use strict;
use warnings;
print("Multiplication tables from 1-10:\n");
sub mul
{
for(my $i=1;$i<=10;$i++)
{
for(my $j=1;$j<=10;$j++)
{
my $result=$i*$j;
print"$i*$j=$result\n";
}
}
}
mul();
```

Output:

```
Multiplication tables from 1-10:
1*1=1
1*2=2
1*3=3
1*4=4
1*5=5
1*6=6
1*7=7
1*8=8
1*9=9
1*10=10
2*1=2
2*2=4
2*3=6
2*4=8
```

2*5=10
2*6=12
2*7=14
2*8=16
2*9=18
2*10=20
3*1=3
3*2=6
3*3=9
3*4=12
3*5=15
3*6=18
3*7=21
3*8=24
3*9=27
3*10=30
4*1=4
4*2=8
4*3=12
4*4=16
4*5=20
4*6=24
4*7=28
4*8=32
4*9=36
4*10=40
5*1=5
5*2=10
5*3=15
5*4=20
5*5=25
5*6=30
5*7=35
5*8=40
5*9=45
5*10=50
6*1=6
6*2=12
6*3=18
6*4=24
6*5=30
6*6=36
6*7=42
6*8=48
6*9=54
6*10=60
7*1=7
7*2=14
7*3=21
7*4=28
7*5=35
7*6=42

```
7*7=49
7*8=56
7*9=63
7*10=70
8*1=8
8*2=16
8*3=24
8*4=32
8*5=40
8*6=48
8*7=56
8*8=64
8*9=72
8*10=80
9*1=9
9*2=18
9*3=27
9*4=36
9*5=45
9*6=54
9*7=63
9*8=72
9*9=81
9*10=90
10*1=10
10*2=20
10*3=30
10*4=40
10*5=50
10*6=60
10*7=70
10*8=80
10*9=90
10*10=100
```

18. Write a Perl program to implement the following list of manipulating functions
a) Shift
b) Unshift
c) Push

Program:

```
#!/usr/bin/perl
use strict;
use warnings;
my @arr=('apple','ball','cat','dog');
print"before using:\n";
print"@arr\n\n";
shift @arr;
print"after using shift:\n";
print"@arr\n\n";
my @arr2=('egg');
unshift @arr2,@arr;
```

```
print"after using unshift:\n";
print"@arr2\n\n";
push @arr,'flight';
print"after using push:\n";
print"@arr\n\n";
```

Output:

```
before using:
apple ball cat dog
after using shift:
ball cat dog
after using unshift:
ball cat dog egg
after using push:
ball cat dog flight
```

19. a) Write a Perl script to substitute a word, with another word in a string.

Program:

```
#!/usr/bin/perl
use strict;
use warnings;
my $string="hello Krupa\n";
$string=~ s/Krupa/sagar/ig;
print $string;
```

Output:

```
hello sagar
```

b) Write a Perl script to validate IP address and email address.

Program for validate IP address :

```
#!/usr/bin/perl
use strict;
use warnings;
print("Enter ip adress:");
my $ip=<stdin>;
chomp($ip);
if($ip =~ m/^(\d\d?\d?)\.(\d\d?\d?)\.(\d\d?\d?)\.(\d\d?\d?)$/)
{
print("ip address-$ip \n");
if($1 <= 255 && $2 <= 255 && $3 <= 255 && $4 <=255)
{
print("each octet of ip address is:",
"whinin the range -$1.$2.$3.$4\n");
print("\n->$ip ip address accepted ! \n");
}
else
{
print("octet(s) out of range.",
"valid num range between 0-225\n");
}
}
else
{
print("ip address $ip is not valid format\n");
}
```

Output:

```
Enter ip adress:12.213.44.66
ip address-12.213.44.66
each octet of ip address is:whinin the range -12.213.44.66
->12.213.44.66 ip address accepted !
Enter ip adress:12.213.54
ip address 12.213.54 is not valid format
```

Program for validate email address:

```
#!/usr/bin/perl
use strict;
use warnings;
use 5.010;
use Email::Valid;
print " Enter an email add";
my $str=<>;
foreach my $email ($str)
{
my $add = Email::Valid->address($email);
say ($add ? "yes '$add'" : "no '$email'");
}
```

Output:

```
Enter an email addkalpana@nn.com
yes 'kalpana@nn.com'
Enter an email add kalpana.com
no ' kalpana.com
```

20. Write a Perl script to print the file in reverse order using command line arguments

Program:

```
perl -ne 'chomp;
> print scalar reverse;'
```

Output:

```
welcome
emoclew
```