1. a) 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 "\n Enter string";**

**str=gets.chomp;**

**puts "\n Enter any number:";**

**n=gets.chomp.to\_i;**

**if(n>0)**

**puts str\*n;**

**else**

**puts "Enter positive number";**

**end**

b) Write a Ruby script which accept the radius of a circle from the user

and compute the perimeter and area.

**PROGRAM**

**puts "Enter radius:\n";**

**rad=gets.chomp.to\_i;**

**pi=3.14;**

**area=pi\*rad\*rad;**

**per=2\*pi\*rad;**

**puts "AREA=#{area} and Perimeter=#{per}";**

2. a) Write a Ruby script which accept the user&#39;s first and last name and

print them in reverse order with a space between them.

**PROGRAM**

**puts "ENTER FIRST NAME:\n";**

**fn=gets.chomp;**

**puts "ENTER LAST NAME:\n";**

**ln=gets.chomp;**

**nf=fn.reverse;**

**nl=ln.reverse;**

**puts "#{nl} #{nf}";**

b) Write a Ruby script to accept a filename from the user print the

extension of that.

**PROGRAM**

**puts "ENTER FILENAME:\n";**

**file=gets.chomp;**

**base=File.basename(file);**

**puts "Basename : #{base}";**

**ext=File.extname(file);**

**puts "Extension : #{ext}";**

**path=File.dirname(file);**

**puts "Pathname : #{path}";**

3. Write a Ruby script to illustrate the use of CGI.

**PROGRAM**

**#!C:\Ruby32-x64\bin\ruby.exe**

**require 'cgi'**

**cgi=CGI.new();**

**puts cgi.header;**

**puts "<html><body>This is a test</body></html>";**

4. Write a Ruby script to illustrate the use of Cookies

C1.rb

#!C:\Ruby32-x64\bin\ruby.exe

require 'cgi'

cgi=CGI.new("html4");

cookie=CGI::Cookie.new('name'=>'MGIT','value'=>'CSE','expires'=>Time.now+3600)

cgi.out('cookie'=>cookie)do

cgi.head + cgi.body {"Cookie stored <a href='c2.rb'>Click here</a>"}

end

C2.rb

#!C:\Ruby32-x64\bin\ruby.exe

require 'cgi'

cgi=CGI.new("html4");

cookie=cgi.cookies['MGIT']

cgi.out('cookie'=>cookie)do

cgi.head + cgi.body{cookie[0]}

end

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

**PROGRAM**

**puts "Enter number 1:\n";**

**p=gets.chomp.to\_i;**

**puts "Enter number 2:\n";**

**q=gets.chomp.to\_i;**

**puts "Enter number 3:\n";**

**r=gets.chomp.to\_i;**

**max=p>q ? (p>r ? p : r):(q>r ? q : r);**

**puts "MAX NUMBER IS :#{max}";**

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

**PROGRAM**

**x=10;**

**while x>0**

**if x%2!=0**

**puts "#{x}";**

**end**

**x=x-1;**

**end**

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

20 otherwise return their sum.

**PROGRAM**

**def check(a,b)**

**if(a==20||b==20)**

**return true;**

**else**

**return a+b;**

**end**

**end**

**q=gets.chomp.to\_i;**

**r=gets.chomp.to\_i;**

**res=check(q,r);**

**puts "#{res}";**

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**

**def check(a,b)**

**if((a<0&&b>100)||(a>100 && b<0))**

**return true;**

**else**

**return false;**

**end**

**end**

**t1=gets.chomp.to\_i;**

**t2=gets.chomp.to\_i;**

**res=check(t1,t2);**

**puts "#{res}";**

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

**arr1=[1,2,3,4,5,6,7,800,900]**

**puts arr1**

**puts "---------------------";**

**arr1.push(999) #adding elements at end fo the array**

**puts arr1**

**puts "---------------------";**

**puts arr1[4]**

**puts "---------------------";**

**puts arr1[-1]**

**puts "---------------------";**

**puts arr1[3,5]**

**puts "---------------------";**

**puts arr1.first(3)**

**puts "---------------------";**

**puts arr1.sort.reverse**

**puts "---------------------";**

**x=[1,2,3]**

**y=[4,5,6]**

**z=x+y**

**puts z;**

**puts "---------------------";**

**puts arr1.delete(3)**

**puts "---------------------";**

**arr1.pop**

**puts arr1.length**

**puts "---------------------";**

**arr1 << 10**

**puts arr1**

**puts "---------------------";**

**arr1.unshift(33)**

**puts arr1**

**puts "---------------------";**

**arr1.insert(2,44)**

**puts arr1**

**puts "---------------------";**

**puts arr1.shift**

10. Write a Ruby program to retrieve the total marks where subject name and

marks of a student stored in a hash.

**PROGRAM**

**marks=Hash.new;**

**marks['C']=gets.chomp.to\_i;**

**marks['Java']=gets.chomp.to\_i;**

**marks['ML']=gets.chomp.to\_i;**

**marks['SL']=gets.chomp.to\_i;**

**tmarks=0;**

**marks.each do |key,value|**

**tmarks+=value;**

**end**

**puts "Total #{tmarks}";**

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

**set i 1;**

**set product 1;**

**gets stdin x;**

**while {$i <= $x} {**

**set product [expr $product\*$i];**

**incr i;**

**}**

puts "Factorial of $x is $product";

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

**proc times\_table {x} {**

**for {set i 1} {$i <= 10} {incr i} {**

**set answer [expr $x\*$i];**

**puts "$x\*$i=$answer";**

**}**

**}**

**proc run\_table {} {**

**gets stdin x;**

**times\_table $x;**

**}**

**run\_table**

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

**set list {a1 b1 A1 c1};**

**puts "$list";**

**set l1 [lsort -ascii $list];**

**puts "$l1";**

**set list1 {1 4 2 6 7 8};**

**set l2 [lsort -integer $list1];**

**puts "$l2";**

**set l3 [lsort -integer -decreasing $list1];**

**puts "$l3";**

**set list3 {3.12 2.09 3.66 8.8};**

**set l4 [lsort -real $list3];**

**puts "$l4";**

**set l5 [lsort -real -decreasing $list3];**

**puts "$l5";**

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

(iii)Traverse the list (iv)Concatenate the list

**set list {1 4 3 2};**

**puts "$list";**

**set list1 {5 7 6 0 9};**

**puts "$list1";**

**set list3 [concat $list $list1]**

**puts "$list3";**

**lappend list1 8 4;**

**puts "$list1";**

**foreach val $list {**

**puts "$val";**

**}**

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

**set now [clock seconds];**

**set filename "second.tcl";**

**set timestamp [file mtime $filename];**

**puts "$timestamp";**

**puts "\n:[clock format $timestamp]";**

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

**puts "Enter file name:";**

**gets stdin file1;**

**set fp [open $file1 r];**

**set data [read $fp];**

**gets stdin file2;**

**set fp1 [open $file2 w+]**

**puts $fp1 $data;**

**close $fp1;**

**puts "File copied";**

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

**use strict;**

**use warnings;**

**my $p=<STDIN>;**

**my $q=<STDIN>;**

**my $r=<STDIN>;**

**my $res=$p>$q?($p>$r?$p:$r):($q>$r?$q:$r);**

**print "Larger is $res";**

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

subroutines. use strict;

use warnings;

sub mul

{

my $n=@\_[0];

print "Multiplication table $n\n";

my $i=1;

my $res=0;

while($i<=10)

{

$res=$n\*$i;

print "$n\*$i=$res\n";

$i++;

}

print "==================";

}

my $x=1;

while($x<=4)

{

&mul($x);

$x++;

}

18. Write a Perl program to implement the following list of manipulating

functions.

a) Shift b) Unshift c) Push

**use strict;**

**use warnings;**

**my @list = (1,2,3,4);**

**print "@list\n";**

**my $ele=shift @list;**

**print "@list\n";**

**push (@list,5);**

**print "@list\n";**

**pop @list;**

**unshift(@list,10);**

**print "@list";**

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

**use strict;**

**use warnings;**

**my $myString=<STDIN>;**

**my $myWord=<STDIN>;**

**my $myWord1=<STDIN>;**

**my $myCount=$myString=~ s/$myWord/$myWord1/;**

**print "$myCount";**

**print "\n $myString";**

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

**use strict;**

**use warnings;**

**use 5.010;**

**use Email::Address;**

**print "Enter email id:";**

**my $em=<STDIN>;**

**chomp($em);**

**my $addresses=Email::Address->parse($em);**

**print "$addresses";**

**if($addresses==1)**

**{**

**print "\nVALID";**

**}**

**else{**

**print "INVALID";**

**}**

**print "enter ip address";**

**my $ip=<STDIN>;**

**if(($ip =~ /^(\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})$/) && ($1<=255 && $2<=255 && $3<=255 && $4<=255))**

**{**

**print "ip address is $ip" ;**

**}**

**else**

**{**

**print "wrong ip address";**

**}**

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

arguments.

**use strict;**

**use warnings;**

**my $file=$ARGV[0];**

**chomp($file);**

**open(DATA,$file) or die $!;**

**my @lines=<DATA>;**

**my @rline=reverse(@lines);**

**print "@rline\n";**

**foreach my $x (@rline)**

**{**

**my $rlines=reverse($x);**

**print "$rlines";**

**}**