Scilab Textbook Companion for Unix: Concepts And Applications by S. Das[1]

Created by
Pranav Bhat T
Unix Programming and Networking
Computer Engineering
National Institute of Technology Karnataka
College Teacher
Prof. B. R. Chandavarkar
Cross-Checked by
Lavitha Pereira

May 23, 2016

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website http://scilab.in

Book Description

Title: Unix: Concepts And Applications

Author: S. Das

Publisher: Tata McGrawhill Education Pvt. Ltd.

Edition: 4

Year: 2006

ISBN: 978-0-07-063546-3

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

List of Scilab Codes

Chapter 1

Getting Started

Scilab code Exa 1.1 Date and Time

```
1  clear()
2  dt=getdate()
3  //clc()
4  disp("Example 1 : Write a code sequence to display
       the current date and time")
5  printf("\n
       **************************************************\
       n")
6  disp("Answer :  ")
7  printf("\n")
8  printf("The current date is %s and the current time
       is %d : %d : %d \nThe day is ",date(),dt(7),dt(8)
       ,dt(9))
9  select dt(5)
10     case 1 then
11     printf("Sunday")
12     case 2 then
13     printf("Monday")
14     case 3 then
15     printf("Tuesday")
16     case 4 then
```

```
17      printf("Wednesday")
18      case 5 then
19      printf("Thursday")
20      case 6 then
21      printf("Friday")
22      case 7 then
23      printf("Saturday")
24  end
25  printf(".")
26  printf("\n
    **********************************************************\
    n")
```

Scilab code Exa 1.2 Clearing the Screen

```
1  clear()
2  //clc()
3  disp("Example 2 :          Clear the current session
       window at the press of a key")
4  printf("\n
    **********************************************************\
    n")
5  disp("Answer  :")
6  disp("")
7  disp("Press any key to clear the Session window ")
8  halt("")
9  //clc()
```

Scilab code Exa 1.3 Calendar display

```
1  clear()
2  //clc()
3  printf("Example 3 :       Display the calendar of the
       current month and of a date \nentered by the
      user ")
4  printf("\n
      **************************************************\
      n")
5  printf("Answer  :\n\n\n")
6  printf("The current date is %s whose calendar is ",
      date())
7  ct=calendar()
8  clc(19)
9  disp(ct(1))
10 disp(ct(2))
11 disp(ct(3))
12 printf("\n
      ------------------------------------------------------------------
      n")
13 printf("Enter a date whose calendar is to be
      displayed \n")
14 x=input('Enter in the format [dd,mm,yyyy],ALONG WITH
       THE PARENTHESIS     ')
15 ct=calendar(x(3),x(2))
16 clc(21)
17 disp(ct(1))
18 disp(ct(2))
19 disp(ct(3))
```

```
20 printf("\n\nEntered Date is %d - %d - %d \n",x(1),x
   (2),x(3))
21 printf("\n
   **********************************************************
   ")
```

Scilab code Exa 1.4 Users present

```
1  clear
2  clc
3
4  disp("Example 4    :                                Display
   all the current users in the Current Unix Session
   ")
5  disp('')
6  printf("\n
   **********************************************************
   n")
7  disp("Answer    :")
8  printf("THE FOLLOWING LINES OF CODE RUN \nONLY IN
   SCILAB INSTALLED IN UNIX ENVIRONMENT.....")
9  printf("\nTHE CONSOLE GETS EXITED IN OTHER\n
   OPERATING SYSTEMS")
10 if(getos() ~= "Linux" )then
11     disp("")
12     halt('Press any key to end the script since the
          OS is not Linux')
13     printf("Close the Scilab Console?....\ny :Yes\
          nAny other key:No")
14     st = input('','s')
15     clc(1)
```

```
16      if( st == "y") then
17          exit
18          end
19  else
20      unix_w("who")
21  end
22  printf("\n
    **********************************************************************
    n")
```

Scilab code Exa 1.5 Processes

```
1  clear
2  clc
3
4  disp("Example 5:  Display all the current working
    processes in the current session")
5  printf("\n
    **********************************************************************
    n")
6  disp('Answer     : ')
7  disp('')
8  halt('Press Enter to display the processes')
9  if (getos()=='Windows') then
10 clc(1)
11 powershell('ps')
12 else
13  clc(1)
14  unix_w('ps')
15 end
16 printf("\n
```

```
    ************************************************************************
    n")
```

Scilab code Exa 1.6 Listing Files

```
clear
mode(-1)
clc

disp("Example 6  :  Display all the files in the
   current directory and files beginning with —>Ex<—
   ")
printf("\n
   ************************************************************************
   n")
disp('Answer     :  ')
halt('Press [Enter] to continue')
disp('Files in the current directory ')
mode(0)
ls
mode(-1)
halt('Press Enter to see files beginning with —>Ex<—
   ')
printf("\n
   ————————————————————————————————————————
   n")
disp('Files Beginning with —>Ex<—')
mode(0)
ls Ex*
mode(-1)
printf("\n
```

```
**********************************************************************
n")
```

Scilab code Exa 1.7 Number of files Unix only

```
1
2  clear
3  clc
4
5  disp("Example 6  :  Display the number of files in
       the current directory ")
6  printf("\n
       **********************************************************************
       n")
7  disp("Answer    :")
8  printf("THE FOLLOWING LINES OF CODE RUN \nONLY IN
       SCILAB INSTALLED IN UNIX ENVIRONMENT.....")
9  printf("\nTHE CONSOLE GETS EXITED IN OTHER\n
       OPERATING SYSTEMS")
10 if(getos() ~= "Linux" )then
11     disp("")
12     halt('Press any key to end the script since the
          OS is not Linux')
13     printf("Close the Scilab Console?....\ny :Yes\
          nAny other key:No")
14     st = input('','s')
15     clc(1)
16     if( st == "y") then
17         exit
18         end
19 else
```

```
20      unix_w("ls|wc")
21  end
22  printf("\n
    **********************************************************************
    n")
```

Scilab code Exa 1.8 Programming basics

```
1  clear
2  clc
3  disp("Example 8: Write a code-sequence to find show
       primitive programming ")
4  printf("\n
    **********************************************************************
    n")
5  disp("Answer    :  ")
6  disp("The following programme takes an input from
       the user adds 2 to it and displays the result")
7  disp("")
8  halt("Ready???...Press Enter to continue")
9  a=input("Enter any number  :  ")
10 clc(1)
11 printf("\n\nThe new result is %d",a+2)
12 printf("\n
    **********************************************************************
    n")
```

Scilab code Exa 1.9 Exitting

```
1  clear
2  clc
3  disp("Example 9: Write a code-sequence to exit the
       console")
4  printf("\n
       **********************************************************************
       n")
5  disp("Answer    :   ")
6  halt("Ready???...Press Enter to continue")
7      printf("Close the Scilab Console?....\ny :Yes\
         nAny other key:No")
8      st = input('','s')
9      clc(1)
10     if( st == "y") then
11         exit
12         end
13 printf("\n
       **********************************************************************
       n")
```

Chapter 2

The Unix Architecture and Command Usage

Scilab code Exa 2.1 Shell Name

```
1
2 clear()
3 clc
4
5 disp('Example 1    :
     Display the current working shell ')
6 disp('
     ************************************************************
     ')
7 disp('Answer     : ')
8 printf('The current Working Shell is ')
9 if (getos()=='Linux') then
10     unix_w("echo $SHELL")
11 else
12     printf("%s",getshell())
13 end
14 disp('
     ************************************************************
```

```
    ')
```

Scilab code Exa 2.2 Type of keywords

```
1  //Program for example 2 chapter 2
2  clear
3  clc
4
5  disp("Example 2: Display the type of a given
       variable or a command ")
6  disp('')
7  printf("\n
       **********************************************************************
       n")
8  disp("Answer    :")
9  printf("THE FOLLOWING LINES OF CODE RUN \nONLY IN
       SCILAB INSTALLED IN UNIX ENVIRONMENT.....")
10 printf("\nTHE CONSOLE GIVES DIFFERENT \nOUTPUT IN
       OTHER OPERATING SYSTEMS")
11 if(getos() ~= "Linux" )then
12     ctd=input("Enter the command or variable whose
          type is to be determined  ")
13     clc(1)
14     pt = input("Enter the command again to confirm  "
          ,"s")
15     clc(1)
16      printf("Continue?....\ny :Yes\nAny other key:No"
          )
17      st = input('','s')
18      clc(2)
19      if( st ~= "y") then
```

```
20          exit
21      else
22          n=type(ctd)
23          clc(1)
24          printf("%s is a ",pt)
25          select n
26  case             1 then
27  printf("a real or complex matrix of double.")
28
29  case             2 then
30  printf('a polynomial matrix.')
31  case            4 then
32  printf('a boolean matrix.')
33  case            5 then
34  printf('a sparse matrix.')
35
36  case 6 then
37  printf('a sparse boolean matrix.')
38
39  case 7 then
40  printf('Matlab sparse matrix')
41  case 8 then
42  printf('a matrix of integers stored on 1 (int8), 2 (
       int16) or 4 (int32) bytes.')
43  case 9 then
44  printf('a matrix of graphic handles.')
45  case 10 then
46  printf('a matrix of character strings.')
47  case 11 then
48  printf('an un-compiled function . A function created
        with deff with argument [n].')
49  case 13 then
50  printf('a compiled function .')
51  case 14 then
52  printf('a function library.')
53  case 15 then
54  printf('a list.')
55  case 16 then
```

```
56 printf('a typed list (tlist).')
57 case 17 then
58 printf('a matrix oriented typed list (mlist).')
59 case 128 then
60 printf('a pointer (Use case: lufact).')
61 case 129 then
62 printf('a size implicit polynomial used for indexing
   .')
63 case 130 then
64 printf('a built-in Scilab function, called also
   gateway (C, C++ or Fortran code).')
65 case 0 then
66 printf('a null variable. It is mainly used
   internally by Scilab. If a function has no
   declared returned argument like disp when it is
   called it returns a null variable. If a function
   is called with an omitted argument in the
   argument list like foo(a,,b) the missing argument
    is assigned to a null variable.')
67
68         end
69         end
70 else
71     disp("Enter the file whose type is to be found")
72     unix_w('read xtun;type $xtun')
73 end
74 printf("\n
   **********************************************************
   n")
```

Scilab code Exa 2.3 Path Variable

```
clear
clc

disp("Example   3 :  Display the value of the path
   variable of the current command interpreter")
disp('
   ************************************************************
   ')
disp("Answer    :")
printf("THE FOLLOWING LINES OF CODE RUN \nONLY IN
   SCILAB INSTALLED IN UNIX ENVIRONMENT.....")
printf("\nTHE CONSOLE GIVES DIFFERENT \nOUTPUT IN
   OTHER OPERATING SYSTEMS\n\n")
halt('Continue  ?? ? ...')
if (getos() ~= "Linux" ) then
    printf("The value of PATH variable is %s",
       SCIHOME)
else
    unix_w('echo The value of path variable is $PATH
       ')
end
disp("
   ************************************************************
   ")
```

Scilab code Exa 2.4 Option using

```
clear
clc

printf('Example 4    :  Show the usage of options in
```

```
    commands by \n\t\ttaking long listing(or detailed
     file listing)) ls -l as an example')
disp('
    **********************************************************************
    ')
disp("Answer      :  ")
halt('Continue  ?? ? ...')
if (getos() ~= "Linux" ) then
    printf("The details of files  in the current
       directory are \n")
    powershell('ls')
else
    unix_w('echo The details of files in the current
        directory are ; ls -l')
end
disp("
    **********************************************************************
    ")
```

Scilab code Exa 2.5 Man Pages

```
clear
clc

disp("Example 5:  Display all the manual pages
    related to a particular entered keyword ")
printf("\n
    **********************************************************************
    n")
disp('Answer     : ')
disp('')
```

```
8  printf("THE FOLLOWING LINES OF CODE RUN \nONLY IN
     SCILAB INSTALLED IN UNIX ENVIRONMENT.....")
9  printf("\nTHE CONSOLE GIVES DIFFERENT \nOUTPUT IN
     OTHER OPERATING SYSTEMS\n\n")
10 halt('Continue  ?? ? ...')
11 if (getos() ~= "Linux" ) then
12     st=input("Enter the keyword to be found in the
         HELP pages   : ","string")
13     clc(1)
14     apropos(st)
15 else
16     disp("Enter the keyword whose man pages are to
         be shown  : ")
17     unix_w('read stx;man $stx')
18 end
19 disp("
    ***********************************************************************
    ")
```

Chapter 3

General Purpose Utilities

Scilab code Exa 3.1 Calendar

```
1  clear()
2  //clc()
3
4  printf("Example 1 :       Display the calendar of the
       current month and of a date \nentered by the
      user ")
5  printf("\n
      **************************************************\
      n")
6  printf("Answer  :\n\n\n")
7  printf("The current date is %s whose calendar is ",
      date())
8  ct=calendar()
9  clc(19)
10 disp(ct(1))
11 disp(ct(2))
12 disp(ct(3))
13 printf("\n
      ------------------------------------------------------------------------\
      n")
14 printf("Enter a date whose calendar is to be
```

```
    displayed \n")
15 x=input('Enter in the format [dd,mm,yyyy],ALONG WITH
    THE PARENTHESIS    ')
16 ct=calendar(x(3),x(2))
17 clc(21)
18 disp(ct(1))
19 disp(ct(2))
20 disp(ct(3))
21 printf("\n\nEntered Date is %d - %d - %d \n",x(1),x
    (2),x(3))
22 printf("\n
    ************************************************
    ")
```

Scilab code Exa 3.2 Date and Time

```
1  clear()
2  dt=getdate()
3  //clc()
4
5  disp("Example 2 : Write a code sequence to display
    the current date and time")
6  printf("\n
    ************************************************\
    n")
7  disp("Answer :  ")
8  printf("\n")
9  printf("The current date is %s and the current time
    is %d : %d : %d \nThe day is ",date(),dt(7),dt(8)
    ,dt(9))
10 select dt(5)
```

```
11      case 1 then
12      printf("Sunday")
13      case 2 then
14      printf("Monday")
15      case 3 then
16      printf("Tuesday")
17      case 4 then
18      printf("Wednesday")
19      case 5 then
20      printf("Thursday")
21      case 6 then
22      printf("Friday")
23      case 7 then
24      printf("Saturday")
25  end
26  printf(".")
27  printf("\n
    **********************************************************\
    n")
```

Scilab code Exa 3.3 Use of echo

```
1  //Program for example 3 chapter 3
2  clear
3  clc
4
5  disp("Example 3:Display a message or the value of a
       variable using the command echo")
6  disp("
    **********************************************************
    ")
```

```
7  disp("Answer     :    ")
8  disp('')
9  halt('Continue ...??   ')
10 if (getos()~='Linux') then
11     x=input('Enter any value to be Displayed Again
           : ','s')
12     clc(1)
13     printf("The entered value is %s",x)
14     printf("\nTo display a Message  :\nEnter
          Filename   ")
15 else
16     unix_w('echo Enter n;read n;echo $n is entered
          value')
17     unix_w('echo Enter filename \c')
18
19 end
20 disp('
   **********************************************************************
   ')
```

Scilab code Exa 3.4 printf usage

```
1  //Program for example 4 chapter 3
2  clear
3  clc
4
5  disp("Example 4:Display a message or the value of a
      variable using the command printf")
6  disp("
   **********************************************************************
   ")
```

```
7 disp("Answer    :   ")
8 disp('')
9 halt('Continue ...??   ')
10 if (getos()~='Linux') then
11      printf("\nTo display a Message  :\nEnter
           Filename   ")
12      printf("\nTo Display the value of a Variable :
           \n My Current Shell is %s\n",getshell())
13 else
14     unix_w('echo No Filename entered\nMy current
          shell is $SHELL\n')
15
16 end
17 disp('
   **********************************************************************
   ')
```

Scilab code Exa 3.5 hexadecimal printing

```
1 //Program for example 5 chapter 3
2 clear
3 clc
4
5 disp("Example 3:Display the value of an input letter
    in its octal and hexadecimal equivalent")
6 disp("
   **********************************************************************
   ")
7 disp("Answer    :   ")
8 disp('')
9 halt('Continue ...??   ')
```

```
10      x=input('Enter any value to be Displayed in its
           hexadecimal and octal equivalent  : ')
11      clc(1)
12      printf("The value %d is %o in octal and %x in
           hexadecimal",x,x,x)
13  disp('
    **********************************************************************
    ')
```

Scilab code Exa 3.6 Calculator

```
1  //Program for example 6 chapter 3
2  clear
3  t='y'
4  clc
5
6   disp("Example 6:            Simulate a calculator to
        evaluate mathematical expressions")
7   disp("
       **********************************************************************
       ")
8   disp("Answer      :     ")
9   disp("Continue...???   ")
10  printf("Enter the expressions to be evaluated one
       by one\n")
11  halt("")
12  //clc()
13  printf("Calculator simulation using bc command in
       Unix\n\n")
14  while t=='y'
15      xt=input("Expression  :: ","string")
```

```
16      if xt==”^c” then
17          break
18      end
19      printf(”\n%d \n\n”,evstr(xt))
20  end
21  //clc()
22  printf(” ______

    \n”)
23  printf(”       |       |       |      /\\     |\\     | |
          /        |      |\n”)
24  sleep(300)
25  printf(”       |       |----|    /--\\    | \\    | |<
                |      |\n”)
26  sleep(300)
27  printf(”       |       |       |  /       \\ |     \\| |
         \\         [---]\n”)
28  sleep(300)
29  disp(”Thank You”)
30  disp(”
    **************************************************************
    ”)
```

Scilab code Exa 3.7 Session recording by script

```
1  //Program for example 6 chapter 3
2  clear
3  t='n'
4  clc
5
6   disp(”Example 7:                Record the current
```

```
      session in a file and open the same")
 7  disp("
      ************************************************************
      ")
 8  disp("Answer      :     ")
 9  disp("Continue...???   ")
10  halt("")
11
12  printf("Enter the sequence of instructions to be
      recorded the session\nEnter yes to end the
      session and view result\n")
13  halt('Press Enter to continue')
14 mclose("Sessiont.txt")
15  diary("Sessiont.txt","new")
16  while t~='yes'
17      //clc()
18      xt=input("","string")
19      if(xt=='yes') then
20          t=xt
21          clc(1)
22      end
23
24      clc(1)
25 if(execstr(xt,'errcatch')==0) then
26      v=evstr(xt)
27      clc(1)
28      disp(v(1))
29  elseif(xt~=t) then
30          disp("Wrong Command or Variable")
31       else
32         disp("")
33  end
34  halt("")
35 end
36 diary("Sessiont.txt","close")
37
38 disp("Check for the file Sessiont.txt in the current
      directory...")
```

```
39 halt("")
40 disp("
   **********************************************************************
   ")
```

Scilab code Exa 3.10 Machine Characteristics

```
1  //Program for example 10 chapter 3
2  clear()
3  clc
4
5  disp('Example 8    :

      Display the current machine name ')
6  disp('
   **********************************************************************
   ')
7  disp('Answer     : ')
8  printf('The current machine details are\n ')
9  if (getos()=='Linux') then
10     unix_w("uname ;uname -r")
11 else
12     printf("Operating System   : %s\n Version   :  %s\
          n",getos(),getversion())
13 end
14 disp('
   **********************************************************************
   ')
```

Scilab code Exa 3.11 Terminal Name

```
1 //Program for example 10 chapter 3
2 clear()
3 clc
4
5 disp('Example 9    :

     Display the user terminal details')
6 disp('
     ************************************************************************
     ')
7 disp('Answer     : ')
8 printf('The current terminal details are\n ')
9 if (getos()=='Linux') then
10     unix_w("tty")
11 else
12     printf("The username details are %s\nThe
          terminal file details are %s\n",home,SCI)
13 end
14 disp('
     ************************************************************************
     ')
```

Chapter 4

The File System

Scilab code Exa 4.1 Displaying the home directory

```
1 mode(-1)
2 clear
3 clc
4
5 disp("Example 1    :                Show the path of
     the home directory of the file system")
6 disp("
     ************************************************************
     ")
7 disp("Answer     :    ")
8 disp("INSTRUCTIONS    : ")
9 printf("\nHere all instructions are preloaded in the
      form of a demo\nPRESS ENTER AFTER EACH COMMAND
     to see its RESULT\nPRESS ENTER AFTER EACH RESULT
     TO GO TO THE NEXT COMMAND\n")
10 halt('.............Press [ENTER] to continue.....')
11 halt("")
12 clc
13 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
14 printf("\n\n$ echo $HOME                          #To
```

```
    get the value of the home directory\n\n")
halt(".....# (hit [ENTER] for result)")
clc(1)
printf(" %s \n",home)
halt(".....# (hit [ENTER] for next instruction)")
printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
halt(".....# (hit [ENTER] for result)")
//clc()
printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
sleep(1000)
```

Scilab code Exa 4.2 Current working Directory

```
mode(-1)
clear

pwd
xt=ans
clc

disp("Example 1    :                Show the path of
    the current working directory")
disp("
    ************************************************************
    ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
printf("\nHere all instructions are preloaded in the
```

```
    form of a demo\nPRESS ENTER AFTER EACH COMMAND
    to see its RESULT\nPRESS ENTER AFTER EACH RESULT
    TO GO TO THE NEXT COMMAND\n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
printf("\n\n$ pwd                              #To get the
    value of the home directory\n\n")
halt(".......# (hit [ENTER] for result)")
clc(1)
printf(" %s \n",xt)
halt(".......# (hit [ENTER] for next instruction)")
printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
halt(".......# (hit [ENTER] for result)")
//clc()
printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
sleep(1000)
```

Scilab code Exa 4.3 Changing Directory

```
clear
pwd
cwd=ans
mode(-1)
clc

```

```
7  printf("Example 3     :                    Show the use of
       cd by going to the home directory \n")
8  disp("
     ************************************************************
     ")
9  disp("Answer      :    ")
10 disp("INSTRUCTIONS    : ")
11 printf("\nHere all instructions are preloaded in the
       form of a demo\nPRESS ENTER AFTER EACH COMMAND
     to see its RESULT\nPRESS ENTER AFTER EACH RESULT
     TO GO TO THE NEXT COMMAND\n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
16
17 printf("\n\n$ pwd                              #To get the
       value of the current directory\n\n")
18 halt(".......# (hit [ENTER] for result)")
19 clc(1)
20 printf(" %s \n",cwd)
21 halt(".......# (hit [ENTER] for next instruction)")
22
23 printf("\n\n\n$ cd                               #To go to
     the home directory\n\n")
24 halt(".......# (hit [ENTER] for next instruction)")
25 clc(1)
26
27 printf("\n\n\n$ pwd                                    \n\n
     ")
28 halt(".......# (hit [ENTER] for result)")
29 clc(1)
30 cd
31 xt=ans
32 printf(" %s \n",xt)
33 halt(".......# (hit [ENTER] for next instruction)")
34
```

```
35 printf("\n\n\n$ exit          #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
36 halt(".......# (hit [ENTER] for result)")
37 cd(cwd)
38 //clc()
39
40 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
41 sleep(1000)
```

Scilab code Exa 4.4 Creating Directories

```
1  clear
2  mode(-1)
3  rmdir('pis')
4  pwd
5  cwd=ans
6  clc
7
8  printf("Example 4     :                Show the method
     of  creating a directory named pis \n \t\tand
     surf to the same directory \n")
9  disp("
     ************************************************************
     ")
10 disp("Answer     :    ")
11 disp("INSTRUCTIONS    : ")
12 printf("\nHere all instructions are preloaded in the
     form of a demo\nPRESS ENTER AFTER EACH COMMAND
     to see its RESULT\nPRESS ENTER AFTER EACH RESULT
```

```
    TO GO TO THE NEXT COMMAND\n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")

printf("\n\n$ mkdir  pis                              #To
    make a directory named pis\n\n")
halt(".......# (hit [ENTER] for result)")
clc(1)
mkdir('pis')

printf("\n\n\n$ cd  pis                             #To go
    to the current directory\n\n")
halt(".......# (hit [ENTER] for next instruction)")
clc(1)
cd 'pis'
xt=ans

printf("\n\n\n$ pwd                                    \n\n
    ")
halt(".......# (hit [ENTER] for result)")
clc(1)
printf(" %s \n",xt)
halt(".......# (hit [ENTER] for next instruction)")

printf("\n\n\n$ exit        #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
halt(".......# (hit [ENTER] for result)")
cd(cwd)
//clc()

printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
sleep(1000)
```

Scilab code Exa 4.5 Removing Directories

```
1  clear
2  mode(-1)
3  rmdir('pis')
4  pwd
5  cwd=ans
6  clc
7
8  printf("Example 5     :                    Show the method
       of  removing a directory\n named pis(after
      creating a directory named pis) \n \t\tand surf
      to the same directory \n")
9  disp("
      ************************************************************
      ")
10 disp("Answer     :    ")
11 disp("INSTRUCTIONS    : ")
12 printf("\nHere all instructions are preloaded in the
       form of a demo\nPRESS ENTER AFTER EACH COMMAND
      to see its RESULT\nPRESS ENTER AFTER EACH RESULT
      TO GO TO THE NEXT COMMAND\n")
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
17
18 printf("\n\n$ mkdir  pis                              #To
      make a directory named pis\n\n")
```

```
19 halt("........# (hit [ENTER] )")
20 clc(1)
21 mkdir('pis')
22
23 printf("\n\n\n$ cd pis;pwd                          #To
      go to the  directory named pis and open it\n\n")
24 halt("........# (hit [ENTER] )")
25 clc(1)
26 cd 'pis'
27 xt=ans
28 printf(" %s \n",xt)
29 halt("........# (hit [ENTER] )")
30
31 printf("\n\n\n$ cd  ..                          #To go
      to the parent directory\n\n")
32 halt("........# (hit [ENTER] )")
33 clc(1)
34 cd(cwd)
35 xt=ans
36
37 printf("\n\n\n$ pwd                                \n\n
      ")
38 halt("........# (hit [ENTER] for result)")
39 clc(1)
40 printf(" %s \n",xt)
41 halt("........# (hit [ENTER] for next instruction)")
42
43
44 printf("\n\n\n$ rmdir pis                            #
      To make a directory named pis     \n\n")
45 halt("........# (hit [ENTER] )")
46 clc(1)
47 rmdir pis
48
49 printf("\n\n\n$ ls  pis                          #To go
      to the current directory\n\n")
50 halt("........# (hit [ENTER] )")
51 clc(1)
```

```
52 ls pis
53 printf(" pis : No such file or directory \n ")
54
55 printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
56 halt(".......# (hit [ENTER] )")
57 cd(cwd)
58 //clc()
59
60 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
61 sleep(1000)
```

Scilab code Exa 4.6 Relative Pathnames

```
1 clear
2 mode(-1)
3 pwd
4 cwd=ans
5 t=tokens(cwd,'\')
6 par=strcat(t(1:(size(t,'r')-1)),'\')
7 clc
8
9 printf("Example 6     :                Show the use of
       relative pathname .. by going to the parent of
      the current directory \n")
10 disp("
      ************************************************************
      ")
11 disp("Answer     :    ")
```

```
12 disp("INSTRUCTIONS    : ")
13 printf("\nHere all instructions are preloaded in the
       form of a demo\nPRESS ENTER AFTER EACH COMMAND
     to see its RESULT\nPRESS ENTER AFTER EACH RESULT
     TO GO TO THE NEXT COMMAND\n")
14 halt('.............Press [ENTER] to continue.....')
15 halt("")
16 clc
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
18
19 printf("\n\n$ pwd                          #To get the
       value of the current directory\n\n")
20 halt(".......# (hit [ENTER] for result)")
21 clc(1)
22 printf(" %s \n",cwd)
23 halt(".......# (hit [ENTER] for next instruction)")
24
25 printf("\n\n\n$ cd  ..                          #To go
     to the current directory\n\n")
26 halt(".......# (hit [ENTER] for next instruction)")
27 clc(1)
28 cd(par)
29
30 printf("\n\n\n$ pwd                                \n\n
     ")
31 halt(".......# (hit [ENTER] for result)")
32 clc(1)
33 printf(" %s \n",par)
34 halt(".......# (hit [ENTER] for next instruction)")
35
36 printf("\n\n\n$ exit        #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
37 halt(".......# (hit [ENTER] for result)")
38 cd(cwd)
39 //clc()
40
```

```
41 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
42 sleep(1000)
```

Scilab code Exa 4.7 ls command

```
1  mode(-1)
2  clear
3  clc
4
5  disp("Example 7     :                    Show the use of
       the file command ls for long listing of files")
6  disp("
       ************************************************************
       ")
7  disp("Answer      :    ")
8  disp("INSTRUCTIONS    : ")
9  printf("\nHere all instructions are preloaded in the
       form of a demo\nPRESS ENTER AFTER EACH COMMAND
       to see its RESULT\nPRESS ENTER AFTER EACH RESULT
       TO GO TO THE NEXT COMMAND\n")
10 halt('.............Press [ENTER] to continue.....')
11 halt("")
12 clc
13 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
       PRELOADED COMMANDS)\n\n\n")
14 printf("\n\n$ ls                              #To get the
       files and directories present in the current
       directory\n\n")
15 halt(".........# (hit [ENTER] for result)")
16 mode(0)
```

```
17 ls
18 mode(-1)
19 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
20 printf("\n\n$ ls  -l                          #To get
     the files and directories present in the current
     directory\n\n")
21 halt("........# (hit [ENTER] for result)")
22 mode(0)
23 powershell('ls')
24 mode(-1)
25 halt("........# (hit [ENTER] for next instruction)")
26 printf("\n\n\n$ exit          #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
27 halt("........# (hit [ENTER] for result)")
28 //clc()
29 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
30 sleep(1000)
```

Chapter 5

Handling Ordinary Files

Scilab code Exa 5.1 cat command

```
1 clear
2 flag=1
3 mode(-1)
4 clc
5
6 printf("Example 1 : Show the method
    of file handling using the cat command \n")
7 disp("
   ************************************************************
   ")
8 disp("Answer : ")
9 disp("INSTRUCTIONS : ")
10 printf("\nHere all instructions are preloaded in the
    form of a demo\nPRESS ENTER AFTER EACH COMMAND
   to see its RESULT\nPRESS ENTER AFTER EACH RESULT
   TO GO TO THE NEXT COMMAND\n")
11 halt('.....Press [ENTER] to continue.....')
12 halt("")
13 clc
14 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
   PRELOADED COMMANDS)\n\n\n")
```

```
15
16
17 printf("\n# Enter the name of the file which you
      want to access  \n\n")
18 nam=input('$ cat  ','s')
19 printf("# This searches for a file named %s \n\n",
      nam)
20
21
22 if ~isfile(nam) then
23     flag=0
24     printf("\n%s : file not found \n",nam)
25     printf("# Create a new file named %s?\n # y :
         Yes \n # n : No \n",nam)
26     resp=input('','s')
27     if resp=='y' then
28         flag=1
29         printf("\n#***Enter the contents of the file
             %s*****\n# [Enter ^ in a newline to end
            and close the file]\n",nam)
30         printf('\n\n$ cat > %s
            #to create a file named %s and fill its
            contents\n',nam,nam)
31         fhdr=mopen(nam,'wt')
32         i=1
33         while %t
34             cont=input(string(i)+'.  ','s')
35             if (cont=='^') then
36                 break
37             end
38             mfprintf(fhdr,"%s\n",cont)
39             i=i+1
40         end
41         mclose(fhdr)
42     end
43 end
44
45 if flag==1 then
```

```
46      i=1
47      clc
48      printf("\n                ==========> %s
          <==========\n\n\n",nam)
49      fhdr=mopen(nam,'rt')
50      while %t
51          [n,a]=mfscanf(fhdr,"%c")
52          if meof(fhdr) then
53              break
54          end
55          printf("%c",a)
56          i=i+1
57      end
58      mclose(fhdr)
59      printf("\n\n%d characters present in the file.\n
          [hit ENTER to continue]\n",i)
60      halt('')
61  else
62      printf("\n\n# file %s is not found and not
          created also\n",nam)
63  end
64
65
66  printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
67  halt("........# (hit [ENTER] for result)")
68  //clc()
69
70  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
71  sleep(1000)
```

Scilab code Exa 5.2 cp command

```
1  clear
2  mode(-1)
3  flag=1
4  pwd
5  xt=ans
6  flag=1
7  clc
8
9  printf("Example 2     :                    Show the method
       of copying files in unix using the cp command \n
   ")
10 disp("
   ************************************************************
   ")
11 disp("Answer     :    ")
12 disp("INSTRUCTIONS    : ")
13 printf("\nHere all instructions are preloaded in the
       form of a demo\nPRESS ENTER AFTER EACH COMMAND
   to see its RESULT\nPRESS ENTER AFTER EACH RESULT
   TO GO TO THE NEXT COMMAND\n")
14 halt('.............Press [ENTER] to continue.....')
15 halt("")
16 clc
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
   PRELOADED COMMANDS)\n\n\n")
18
19 src=input("# Enter the name of the file[or directory
   ] which you want to copy  : ",'s')
```

```
20 if isdir(src) then
21 destn=input("# Enter the name of the directory which
       you want to copy into  : ",'s')
22 else
23     destn=input("# Enter the name of the file[or
          directory] which you want to copy into  : ",'
          s')
24 end
25
26 flag=0
27 printf("\n $ cp %s %s   \t#copies file[or directory]
       contents of %s to %s\n",src,destn,src,destn)
28 halt('')
29
30
31 if isfile(destn)&isfile(src) then
32       printf('cp  : overwrite %s (yes/no)?  ',destn)
33       resp=input(' ','s')
34       if resp=='y' then
35         mdelete(destn)
36       end
37 end
38
39 if isfile(src)|isdir(src) then
40     flag=1
41     [status,msg]=copyfile(src,destn)
42 else
43     printf("\n%s  : file or directory not found \n",
          src)
44     flag=0
45 end
46
47 if flag==1&isfile(destn) then
48     i=1
49     printf("\n $ cat %s \t#to display the copied
          file %s \n\n",destn,destn)
50     printf("\n          ==========> %s
          <==========\n\n\n",destn)
```

```
51      fhdr=mopen(destn,'rt')
52      while %t
53          [n,a]=mfscanf(fhdr,"%c")
54          if meof(fhdr) then
55              break
56          end
57          printf("%c",a)
58          i=i+1
59      end
60      mclose(fhdr)
61      printf("\n\n%d characters present in the file.\n
           [hit ENTER to continue]\n",i)
62      halt('')
63      elseif isdir(destn)&flag==1 then
64      cd(destn)
65      mode(0)
66      ls
67      halt("Go back to previous directory  ?? ")
68      mode(-1)
69      cd(xt)
70  else
71      printf("\n\n# file %s is not rewritten using
           copy command cp and not created also\n",destn
           )
72  end
73
74
75  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
76  halt("........# (hit [ENTER] for result)")
77  //clc()
78
79  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
80  sleep(1000)
```

Scilab code Exa 5.3 rm command

```
1 clear
2 flag=1
3 mode(-1)
4 clc
5
6
7 printf("Example 3     :                    Show the method
      of removing files in unix using the rm command \
     n")
8 disp("
     ************************************************************
     ")
9 disp("Answer      :    ")
10 disp("INSTRUCTIONS     : ")
11 printf("\nHere all instructions are preloaded in the
      form of a demo\nPRESS ENTER AFTER EACH COMMAND
     to see its RESULT\nPRESS ENTER AFTER EACH RESULT
     TO GO TO THE NEXT COMMAND\n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
16 src=input("# Enter the name of the file which you
     want to delete  : ",'s')
17
18 flag=0
19 printf("\n $ rm %s   \t#deletes file %s\n",src,src)
```

```
20 halt('')
21
22
23 if isfile(src) then
24       printf('rm  : remove %s (yes/no)?  ? ',src)
25       resp=input(' ','s')
26       if resp=='y' then
27         mdelete(src)
28         flag=1
29       end
30   else
31     printf("\n%s  : file not found \n",src)
32 end
33
34 if flag then
35     printf("\n $ cat %s              # opening file
          %s to see if it exists \n",src,src)
36     if ~isfile(src) then
37         printf("\n%s : file not found \n ",src)
38     else
39
40     printf("\n           ==========> %s
          <==========\n\n\n",destn)
41     fhdr=mopen(destn,'rt')
42     while %t
43         [n,a]=mfscanf(fhdr,"%c")
44         if meof(fhdr) then
45             break
46         end
47         printf("%c",a)
48         i=i+1
49     end
50     mclose(fhdr)
51     printf("\n\n%d characters present in the file.\n
          [hit ENTER to continue]\n",i)
52     halt('')
53
54     end
```

```
55 end
56
57
58 printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
59 halt("........# (hit [ENTER] for result)")
60 //clc()
61
62 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
63 sleep(1000)
```

Scilab code Exa 5.4 mv command

```
1  clear
2  mode(-1)
3  flag=1
4  pwd
5  xt=ans
6  clc
7
8  printf("Example 4     :                Show the method
       of renaming files in unix using the mv command \
      n")
9  disp("
      ************************************************************
      ")
10 disp("Answer     :    ")
11 disp("INSTRUCTIONS    : ")
12 printf("\nHere all instructions are preloaded in the
```

```
      form of a demo\nPRESS ENTER AFTER EACH COMMAND
     to see its RESULT\nPRESS ENTER AFTER EACH RESULT
     TO GO TO THE NEXT COMMAND\n")
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
17
18 src=input("# Enter the name of the file[or directory
     ] which you want to rename  : ",'s')
19 if isdir(src) then
20 destn=input("# Enter the new name of the directory
     : ",'s')
21 else
22     destn=input("# Enter the name of the file[or
         directory] which you want to move into  : ",'
         s')
23 end
24
25 flag=0
26 printf("\n $ mv %s %s   \t#copies file[or directory]
     contents of %s to %s\n",src,destn,src,destn)
27 halt('')
28
29
30 if isfile(destn)&isfile(src) then
31       printf('mv  : overwrite %s (yes/no)?  ',destn)
32       resp=input(' ','s')
33       if resp=='y' then
34         mdelete(destn)
35       end
36 end
37
38 if isfile(src)|isdir(src) then
39     flag=1
40     [status,msg]=movefile(src,destn)
41 else
```

```
42      printf("\n%s  : file or directory not found \n",
           src)
43      flag=0
44  end
45
46  if flag==1&isfile(destn) then
47      i=1
48      printf("\n $ cat %s \t#to display the moved file
            %s \n\n",destn,destn)
49      printf("\n           ==========> %s
          <==========\n\n\n",destn)
50      fhdr=mopen(destn,'rt')
51      while %t
52          [n,a]=mfscanf(fhdr,"%c")
53          if meof(fhdr) then
54              break
55          end
56          printf("%c",a)
57          i=i+1
58      end
59      mclose(fhdr)
60      printf("\n\n%d characters present in the file.\n
           [hit ENTER to continue]\n",i)
61      halt('')
62  elseif isdir(destn)&flag==1 then
63      cd(destn)
64      mode(0)
65      ls
66      halt("Go back to previous directory  ?? ")
67      mode(-1)
68      cd(xt)
69  else
70      printf("\n # No changes done in the file \n")
71  end
72
73
74  printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
```

```
    n\n")
75 halt("........# (hit [ENTER] for result)")
76 //clc()
77
78 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
79 sleep(1000)
```

Scilab code Exa 5.5 lp command

```
1  mode(-1)
2  clear
3  flag=1
4  clc
5
6  printf("Example 5    :                Show the method
     of file printing using the lp command \n")
7  disp("
    ************************************************************
    ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 printf("\nHere all instructions are preloaded in the
     form of a demo\nPRESS ENTER AFTER EACH COMMAND
    to see its RESULT\nPRESS ENTER AFTER EACH RESULT
    TO GO TO THE NEXT COMMAND\n")
11 halt('.............Press [ENTER] to continue.....')
12 halt("")
13 clc
14 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
```

```
15
16
17 printf("\n# Enter the name of the file which you
      want to print  \n\n")
18 nam=input('$ lp  ','s')
19 printf("# This sends the  file named %s to printer
      and gets status\n\n",nam)
20
21
22 if ~isfile(nam) then
23     flag=0
24     printf("\n%s : file not found \n",nam)
25     printf("# Create a new file named %s?\n # y :
         Yes \n # n : No \n",nam)
26     resp=input('','s')
27     if resp=='y' then
28         flag=1
29         printf("\n#***Enter the contents of the file
             %s*****\n# [Enter ^ in a newline to end
            and close the file]\n",nam)
30         printf('\n\n$ cat > %s
            #to create a file named %s and fill its
            contents\n',nam,nam)
31         fhdr=mopen(nam,'wt')
32         i=1
33         while %t
34             cont=input(string(i)+'.  ','s')
35             if (cont=='^') then
36                 break
37             end
38             mfprintf(fhdr,"%s\n",cont)
39             i=i+1
40         end
41         mclose(fhdr)
42     end
43 end
44 if flag then
45 s=toprint(nam)
```

```
46 if s then
47     printsetupbox()
48 else
49     printf("\n\nlp : printer busy \n")
50 end
51 end
52
53 printf("\n\n\n$ exit          #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
54 halt(".......# (hit [ENTER] for result)")
55 //clc()
56
57 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
58 sleep(1000)
```

Scilab code Exa 5.6 wc command

```
1 mode(-1)
2 clear
3 flag=1
4 clc
5
6 printf("Example 6     :                Show the method
     of file counting using the wc command \n")
7 disp("
     ************************************************************
     ")
8 disp("Answer     :    ")
9 disp("INSTRUCTIONS    : ")
```

```
10 printf("\nHere all instructions are preloaded in the
       form of a demo\nPRESS ENTER AFTER EACH COMMAND
     to see its RESULT\nPRESS ENTER AFTER EACH RESULT
     TO GO TO THE NEXT COMMAND\n")
11 halt('.............Press [ENTER] to continue.....')
12 halt("")
13 clc
14 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
15
16
17 printf("\n# Enter the name of the file which you
     want to access  \n\n")
18 nam=input('$ cat   ','s')
19 printf("# This searches for a file named %s to
     display\n\n",nam)
20
21
22 if ~isfile(nam) then
23     flag=0
24     printf("\n%s : file not found \n",nam)
25     printf("# Create a new file named %s?\n # y :
         Yes \n # n : No \n",nam)
26     resp=input('','s')
27     if resp=='y' then
28         flag=1
29         printf("\n#***Enter the contents of the file
               %s*****\n# [Enter ^ in a newline to end
             and close the file]\n",nam)
30         printf('\n\n$ cat > %s
             #to create a file named %s and fill its
             contents\n',nam,nam)
31         fhdr=mopen(nam,'wt')
32         i=1
33         while %t
34             cont=input(string(i)+'.  ','s')
35             if (cont=='^') then
36                 break
```

```
37              end
38              mfprintf(fhdr,"%s\n",cont)
39              i=i+1
40          end
41          mclose(fhdr)
42      end
43  end
44
45  if flag==1 then
46      c=1
47      w=0
48      l=0
49      clc
50      printf("\n $ cat %s     \n",nam)
51      fhdr=mopen(nam,'rt')
52      while %t
53          [n,a]=mfscanf(fhdr,"%c")
54          if meof(fhdr) then
55              break
56          end
57
58              printf("%c",a)
59              c=c+1
60           if ascii(a)==32 then
61              w=w+1
62          end
63          if ascii(a)==10 then
64              w=w+1
65              l=l+1
66          end
67      end
68      mclose(fhdr)
69      halt('')
70        printf('\n\n$ wc %s                              #to
             get the count in file named %s \n',nam,nam)
71     halt('')
72     printf('\t%d\t%d\t%d    %s\n',l,w,c,nam)
73      printf("\n# This means there are %d words,%d
```

```
        characters\n \t and %d lines in the file %s \
        n",w,c,l,nam)
74      printf('\n\n$ wc -l  %s                    #
          to get the line count in file named %s \n',
          nam,nam)
75    halt('')
76    printf('\t%d  %s\n',l,nam)
77     printf("\n# Number of lines \n")
78      printf('\n\n$ wc -w  %s                    #
          to get the word count in file named %s \n',
          nam,nam)
79    halt('')
80    printf('\t%d  %s\n',w,nam)
81     printf("\n# Number of words \n")
82      printf('\n\n$ wc -c  %s                    #
          to get the character count in file named %s
          \n',nam,nam)
83    halt('')
84    printf('\t%d  %s\n',c,nam)
85     printf("\n# Number of characters \n")
86  end
87
88
89  printf("\n\n\n$ exit        #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
90  halt(".......# (hit [ENTER] for result)")
91  //clc()
92
93  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
94  sleep(1000)
```

Scilab code Exa 5.7 od command

```
clear
flag=1
clc
mode(-1)

printf("Example 7     :                    Show the method
    of file handling using the od command \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
printf("\nHere all instructions are preloaded in the
    form of a demo\nPRESS ENTER AFTER EACH COMMAND
   to see its RESULT\nPRESS ENTER AFTER EACH RESULT
   TO GO TO THE NEXT COMMAND\n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
   PRELOADED COMMANDS)\n\n\n")


printf("\n# Enter the name of the file which you
   want to access  \n\n")
nam=input('$ od  ','s')
printf("# This searches for a file named %s \n\n",
   nam)
```

```
20
21
22 if ~isfile(nam) then
23     flag=0
24     printf("\n%s : file not found \n",nam)
25     printf("# Create a new file named %s?\n # y :
          Yes \n # n : No \n",nam)
26     resp=input('','s')
27     if resp=='y' then
28         flag=1
29         printf("\n#***Enter the contents of the file
              %s*****\n# [Enter ^ in a newline to end
             and close the file]\n",nam)
30         printf('\n\n$ cat > %s
             #to create a file named %s and fill its
             contents\n',nam,nam)
31         fhdr=mopen(nam,'wt')
32         i=1
33         while %t
34             cont=input(string(i)+'.  ','s')
35             if (cont=='^') then
36                 break
37             end
38             mfprintf(fhdr,"%s\n",cont)
39             i=i+1
40         end
41         mclose(fhdr)
42     end
43 end
44
45 if flag==1 then
46     i=1
47     clc
48     printf("\n $ od %s     #to display %s in octal
          characters\n\n",nam,nam)
49     printf("\n          ==========> %s
          <==========\n\n\n",nam)
50     fhdr=mopen(nam,'rt')
```

```
51      while %t
52          [n,a]=mfscanf(fhdr,"%c")
53          if meof(fhdr) then
54              break
55          end
56          printf(" %o",ascii(a))
57          if ascii(a)==10 then
58              printf("\n")
59          end
60          i=i+1
61      end
62      mclose(fhdr)
63      printf("\n\n%d characters present in the file.\n
           [hit ENTER to continue]\n",i)
64      halt('')
65  else
66      printf("\n\n# file %s is not found and not
           created also\n",nam)
67  end
68
69  flag=flag+1
70
71  octs=blanks(0)
72  if flag==2 then
73      i=1
74      clc
75       printf("\n $ od -bc %s      #to display %s in
           octal characters\n\n",nam,nam)
76      printf("\n              ==========> %s
           <==========\n\n\n",nam)
77      fhdr=mopen(nam,'rt')
78      while %t
79          [n,a]=mfscanf(fhdr,"%c")
80          if meof(fhdr) then
81              break
82          end
83          printf("    %c   ",a)
84          octs=octs+string(dec2oct(ascii(a)))+'   '
```

```
85          if ascii(a)==10 then
86              printf("%s\n\n",octs)
87              clear('octs')
88              octs=blanks(0)
89          end
90          i=i+1
91      end
92      mclose(fhdr)
93      printf("\n\n%d characters present in the file.\n
           [hit ENTER to continue]\n",i)
94      halt('')
95  else
96      printf("\n\n# file %s is not found and not
           created also\n",nam)
97  end
98
99  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
100 halt("........# (hit [ENTER] for result)")
101 //clc()
102
103 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
104 sleep(1000)
```

Scilab code Exa 5.8 cmp command

```
1  clear
2  flag=1
3  clc
```

```
4  mode(-1)
5
6  printf("Example 8     :                    Show the method
       of comparing files using cmp command \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer      :    ")
9  disp("INSTRUCTIONS    : ")
10 printf("\nHere all instructions are preloaded in the
       form of a demo\nPRESS ENTER AFTER EACH COMMAND
      to see its RESULT\nPRESS ENTER AFTER EACH RESULT
      TO GO TO THE NEXT COMMAND\n")
11 halt('.............Press [ENTER] to continue.....')
12 halt("")
13 clc
14 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
15
16
17 printf("\n# Enter the name of the two files which
      you want to compare \n")
18 fil(1)=input('','s')
19 fil(2)=input('','s')
20 printf('\n\n $ cmp   %s  %s     #to compare the
      files %s and %s\n',fil(1),fil(2),fil(1),fil(2))
21
22 for i=1:2
23 if ~isfile(fil(i)) then
24     flag(i)=0
25     printf("\n%s : file not found \n",fil(i))
26     printf("# Create a new file named %s?\n # y :
          Yes \n # n : No \n",fil(i))
27     resp=input('','s')
28     if resp=='y' then
29         flag(i)=1
30         printf("\n#***Enter the contents of the file
               %s*****\n# [Enter ^ in a newline to end
```

```
            and close the file]\n",fil(i))
31          printf('\n\n$ cat > %s
               #to create a file named %s and fill its
               contents\n',fil(i),fil(i))
32          fhdr=mopen(fil(i),'wt')
33          count=1
34          while %t
35              cont=input(string(count)+'.  ','s')
36              if (cont=='^') then
37                  break
38              end
39              mfprintf(fhdr,"%s\n",cont)
40              count=count+1
41          end
42          mclose(fhdr)
43      end
44  end
45  end
46
47
48  if flag(1)&flag(2) then
49      clc
50      printf("\n $ cmp   %s  %s    #to compare files
           %s and %s \n",fil(1),fil(2),fil(1),fil(2))
51      fhdr1=mopen(fil(1),'rt')
52      fhdr2=mopen(fil(2),'rt')
53      l=0
54      cr=1
55      while %t
56          [n,a1]=mfscanf(fhdr1,"%c")
57          [n,a2]=mfscanf(fhdr2,"%c")
58          if meof(fhdr1)&meof(fhdr2) then
59              printf("  \n# No output means both the
                   files are identical \n")
60              break
61           elseif (meof(fhdr1)&~meof(fhdr2))|(meof(
               fhdr2)&~meof(fhdr1))|a1~=a2
62            printf(" %s  %s  differ :  char %d ,  line
```

```
                %d \n",fil(1),fil(2),cr,l+1)
63            printf(" # This shows that %dth character
                 in %dth line do not match\n\n",cr,l+1)
64            break
65           end
66          cr=cr+1
67          if ascii(a1)==10 then
68              l=l+1
69              cr=1
70          end
71      end
72      mclose(fhdr1)
73      mclose(fhdr2)
74      halt('')
75  else
76      printf("\n\n# file %s or %s is not found \n",fil
           (1),fil(2))
77  end
78
79
80  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
81  halt("........# (hit [ENTER] for result)")
82  //clc()
83
84  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
85  sleep(1000)
```

Chapter 14

Essential Shell Programming

Scilab code Exa 14.1 Program 1

```
1
2  clear
3  clc
4  mode(-1)
5
6   disp("Example 1: Write a shell script to display
       the calendar of the present date ,and the shell
       name")
7   disp("
       ************************************************************
       ")
8   disp("Answer   :    ")
9   disp("")
10  halt("The code related to the example lies in the
       dependency file shell1.sci    ")
11  printf("Today  date is %s",date())
12  printf("\nThis month calendar is \n")
13  calendar()
14  t=ans
15  disp(t(1))
16  disp(t(2))
```

```
17  disp(t(3))
18  printf("\nMy shell : %s",getshell())
19  disp("
      ************************************************************
      ")
```

Scilab code Exa 14.2 Program 2

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 2     :                Show the method
       of using read to take two inputs \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
```

```
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
16 i=0
17 i=i+1;f(i)='2233|a.k.shukla                |g.m.
            |sales            |12/12/52|6000'
18 i=i+1;f(i)='9876|jai sharma                |director   |
      production |12/03/50|7000'
19 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m       |
      marketing  |19/04/43|6000'
20 i=i+1;f(i)='2356|barun sengupta       |director   |
      personnel  |11/05/47|7800'
21 i=i+1;f(i)='5423|n.k. gupta              |chairman |
      admin       |30/08/56|5400'
22 i=i+1;f(i)='1006|chanchal singhvi    |director   |
      sales          |03/09/38|6700'
23 i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
      accounts   |05/06/62|6300'
24 i=i+1;f(i)='1265|s.n. dasgupta          |manager  |
      sales          |12/09/63|5600'
25 i=i+1;f(i)='4290|jayant Choudhary  |executive |
      production|07/09/50|6000'
26 i=i+1;f(i)='2476|anil aggarwal         |manager    |
      sales          |01/05/59|5000'
27 i=i+1;f(i)='6521|lalit chowdury        |director     |
      marketing |26/09/45|8200'
28 i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
      accounts   |12/12/55|6000'
29 i=i+1;f(i)='3564|sudhir Agarwal      |executive  |
      personnel  |06/07/47|7500'
30 i=i+1;f(i)='2345|j.b. saxena              |g.m.
               |marketing |12/03/45|8000'
31 i=i+1;f(i)='0110|v.k. agrawal          |g.m.
               |marketing  |31/02/40|9000'
32 n=i
33 printf("\n\n$ cat emp.lst      # to open the file
```

```
    emp.lst")
34  halt(' ')
35  u=mopen('emp.lst','wt')
36  for i=1:n
37      mfprintf(u,"%s\n",f(i))
38      printf("%s\n",f(i))
39  end
40  mclose(u)
41  halt('')
42  clc
43  li(1)='#!/bin/sh'
44  li(2)='# emp1.sh : Interactive version - uses read
    to take two inputs'
45  li(3)='#'
46  li(4)='echo '+ascii(34)+'Enter the pattern to be
    searched:   \c '+ascii(34)+'      # No newline'
47  li(5)='read pname'
48  li(6)='echo  '+ascii(34)+'Enter the file to be used:
     \c '+ascii(34)+'           #use echo -e or shopt
     -s xpg_echo in bash'
49  li(7)='read flname'
50  li(8)='echo  '+ascii(34)+'Searching for $pname from
    file $flname '+ascii(34)
51  li(9)='grep  '+ascii(34)+'$pname '+ascii(34)+'
    $flname'
52  li(10)='echo  '+ascii(34)+'Selected records shown
    above '+ascii(34)
53
54  printf("\n# Enter the name of the shellscript file
    whichever you desire  \n\n")
55  nam=input('$ cat  ','s')
56  halt(' ')
57
58  for i=1:10
59      printf("%s\n",li(i))
60  end
61  halt(' ')
62  clc
```

```
63 lst(1)='@echo off'
64 lst(2)='set /P pname=Enter the pattern to be
      searched:  '
65 lst(3)='rem echo.'
66 lst(4)='set /P flname=Enter the file to be used:  '
67 lst(5)='rem echo.'
68 lst(6)='echo Searching for %pname% from file
      %flname%'
69 lst(7)='rem echo.'
70 lst(8)='findstr /C:'+ascii(34)+'%pname%'+ascii(34)+'
      %flname% '
71 lst(9)='echo Selected records shown above'
72 lst(10)='pause>null'
73
74 if getos()=='Linux' then
75     printf("\n\nPlease Switch to windows and then
         execute\n\nThank You \n\n")
76     halt(' ')
77     exit
78 end
79
80
81 v=mopen(nam+'.sh.bat','wt')
82 for i=1:10
83     mfprintf(v,"%s\n",lst(i))
84 end
85 mclose(v)
86
87
88 printf("\n# type the following command in the
      command line interpreter as soon as it appears")
89 printf(" \n       %c  %s.sh     %c[ENTER]\n\n",ascii
      (34),nam,ascii(34))
90
91 printf("\n$ %s.sh                #to execute the
      perlscript",nam)
92
93 halt(' ')
```

```
94 dos('start')
95 printf("\n\n\n")
96 halt('   ------------------>Executing ShellScript in
      Command Line Prompt<------------------ ')
97 printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
98 halt(".......# (hit [ENTER] for result)")
99 //clc()
100
101 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
102 sleep(1000)
103
104 mdelete(nam+'.sh.bat')
105 mdelete('emp.lst')
```

Scilab code Exa 14.3 Program 3

```
1 clear
2 flag=1
3 mode(-1)
4 clc
5
6 printf("Example 3     :                Show the method
     of using command line arguments to take inputs \
     n")
7 disp("
     ************************************************************
     ")
8 disp("Answer     :    ")
```

```
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
16 i=0
17 i=i+1;f(i)='2233|a.k.shukla              |g.m.
           |sales          |12/12/52|6000'
18 i=i+1;f(i)='9876|jai sharma              |director   |
      production |12/03/50|7000'
19 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m      |
      marketing  |19/04/43|6000'
20 i=i+1;f(i)='2356|barun sengupta       |director   |
      personnel  |11/05/47|7800'
21 i=i+1;f(i)='5423|n.k. gupta              |chairman |
      admin      |30/08/56|5400'
22 i=i+1;f(i)='1006|chanchal singhvi    |director   |
      sales          |03/09/38|6700'
23 i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
      accounts   |05/06/62|6300'
24 i=i+1;f(i)='1265|s.n. dasgupta          |manager  |
      sales          |12/09/63|5600'
25 i=i+1;f(i)='4290|jayant Choudhary  |executive |
      production|07/09/50|6000'
26 i=i+1;f(i)='2476|anil aggarwal         |manager    |
      sales          |01/05/59|5000'
```

```
27 i=i+1;f(i)='6521|lalit chowdury        |director     |
     marketing |26/09/45|8200'
28 i=i+1;f(i)='3212|shyam saksena        |d.g.m        |
     accounts    |12/12/55|6000'
29 i=i+1;f(i)='3564|sudhir Agarwal       |executive    |
     personnel   |06/07/47|7500'
30 i=i+1;f(i)='2345|j.b. saxena              |g.m.
                |marketing |12/03/45|8000'
31 i=i+1;f(i)='0110|v.k. agrawal           |g.m.
                |marketing   |31/02/40|9000'
32 n=i
33 printf("\n\n$ cat emp.lst       # to open the file
     emp.lst")
34 halt(' ')
35 u=mopen('emp.lst','wt')
36 for i=1:n
37     mfprintf(u,"%s\n",f(i))
38     printf("%s\n",f(i))
39 end
40 mclose(u)
41 halt('')
42 clc
43 li(1)='#!/bin/sh'
44 li(2)='# emp1.sh : Interactive version - uses read
     to take two inputs'
45 li(3)='#'
46 li(4)='echo '+ascii(34)+'Program:  $0  '+ascii(34)+'
          #$0 has the program name'
47 li(5)='echo  '+ascii(34)+'The number of arguments
     specified is $# '+ascii(34)
48 li(6)='echo  '+ascii(34)+'The arguments are $* '+
     ascii(34) + '  #All arguments in $*'
49 li(7)='grep  '+ascii(34)+'$1'+ascii(34)+'$2'
50 li(8)='echo  '+ascii(34)+'\nJob Over '+ascii(34)
51
52 printf("\n# Enter the name of the shellscript file
     whichever you desire  \n\n")
53 nam=input('$ cat  ','s')
```

```
54 halt(' ')
55
56 for i=1:8
57     printf("%s\n",li(i))
58 end
59 halt(' ')
60 clc
61 lst(1)='@echo off'
62
63
64 lst(2)='echo Program:  '+nam+'.sh'
65 lst(3)='set a=0'
66 lst(4)='for %%x in (%*) do set /A a+=1'
67 lst(5)='echo The number of arguments specified is
     %a%'
68 lst(6)='echo The arguments are %*'
69 lst(7)='findstr /C:'+ascii(34)+'%1'+ascii(34)+' %2'
70 lst(8)='echo.'
71 lst(9)='echo Job Over'
72 lst(10)='pause>null'
73
74 if getos()=='Linux' then
75     printf("\n\nPlease Switch to windows and then
         execute\n\nThank You \n\n")
76     halt(' ')
77     exit
78 end
79
80
81
82
83 v=mopen(nam+'.sh.bat','wt')
84 for i=1:10
85     mfprintf(v,"%s\n",lst(i))
86 end
87 mclose(v)
88
89
```

```
90 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
91 printf(" \n       %c  %s.sh     %c [COMMANDLINE
     ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
92
93 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
94
95 halt(' ')
96 dos('start')
97 printf("\n\n\n")
98 halt('  ------------------>Executing ShellScript in
     Command Line Prompt<---------------- ')
99 printf("\n\n\n$ exit          #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
100 halt("........# (hit [ENTER] for result)")
101 //clc()
102
103 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
104 sleep(1000)
105
106 mdelete(nam+'.sh.bat')
107 mdelete('emp.lst')
```

Scilab code Exa 14.4 Program 4

```
1 clear
2 flag=1
3 mode(-1)
```

```
4  clc
5
6  printf("Example 4     :                Show the method
       of using if else construct in shell progamming \
      n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
16
17 halt('')
18 clc
19 li(1)='#!/bin/sh'
20 li(2)='# emp3.sh : Using if and else'
21 li(3)='#'
22 li(4)='if grep '+ascii(34)+'^$1'+ascii(34)+' /etc/
      passwd 2> /dev/null           #  Search username
      at beginning of line'
23 li(5)='then'
24 li(6)='          echo  '+ascii(34)+' Pattern found
      - Job Over '+ascii(34)
```

```
li(7)='else'
li(8)='            echo  '+ascii(34)+' Pattern not
   found '+ascii(34)
li(9)='fi'
printf("\n# Enter the name of the shellscript file
   whichever you desire  \n\n")
nam=input('$ cat  ','s')
halt(' ')

for i=1:9
    printf("%s\n",li(i))
end
halt(' ')
clc
lst(1)='@echo off&&cls'
lst(2)='dir /b \Users>passwd'
lst(3)='findstr /b '+ascii(34)+'%1'+ascii(34)+'
   passwd > tmpfil '
lst(4)='set a=tmpfil '
lst(5)='for /F  '+ascii(34)+'usebackq '+ascii(34)+'
   %%A in ( '+ascii(39)+'%a% '+ascii(39)+') do set
   y=%%~zA'
lst(6)='if %y% neq 0 (echo Pattern Found - Job Over)
    else (echo Pattern not found )'
lst(7)='pause>nul'
lst(8)='del tmpfil '
lst(9)='del passwd'

if getos()=='Linux' then
    printf("\n\nPlease Switch to windows and then
       execute\n\nThank You \n\n")
    halt(' ')
    exit
end


v=mopen(nam+'.sh.bat','wt')
for i=1:9
```

```
56      mfprintf(v,"%s\n",lst(i))
57  end
58  mclose(v)
59
60
61  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
62  printf(" \n        %c  %s.sh      %c [COMMANDLINE
       ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
63
64  printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                  #to execute the perlscript",nam)
65
66  halt(' ')
67  dos('start')
68  printf("\n\n\n")
69  halt('  ------------------>Executing ShellScript in
       Command Line Prompt<---------------- ')
70  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
71  halt("........# (hit [ENTER] for result)")
72  //clc()
73
74  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
75  sleep(1000)
76
77  mdelete(nam+'.sh.bat')
78  mdelete('emp.lst')
```

Scilab code Exa 14.5 Program 5

```
clear
flag=1
mode(-1)
clc

printf("Example 5     :                    Show the method
    of using if-elif construct in shell \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
   AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
   ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
   \n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
   PRELOADED COMMANDS)\n\n\n")
i=0
i=i+1;f(i)='2233|a.k.shukla              |g.m.
          |sales          |12/12/52|6000'
i=i+1;f(i)='9876|jai sharma              |director   |
   production |12/03/50|7000'
i=i+1;f(i)='5678|sumit chakrobarty|d.g.m      |
   marketing  |19/04/43|6000'
i=i+1;f(i)='2356|barun sengupta       |director   |
```

```
    personnel  |11/05/47|7800'
21  i=i+1;f(i)='5423|n.k. gupta              |chairman |
    admin       |30/08/56|5400'
22  i=i+1;f(i)='1006|chanchal singhvi    |director  |
    sales          |03/09/38|6700'
23  i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
    accounts   |05/06/62|6300'
24  i=i+1;f(i)='1265|s.n. dasgupta          |manager  |
    sales          |12/09/63|5600'
25  i=i+1;f(i)='4290|jayant Choudhary  |executive |
    production|07/09/50|6000'
26  i=i+1;f(i)='2476|anil aggarwal         |manager   |
    sales          |01/05/59|5000'
27  i=i+1;f(i)='6521|lalit chowdury        |director    |
    marketing |26/09/45|8200'
28  i=i+1;f(i)='3212|shyam saksena       |d.g.m       |
    accounts   |12/12/55|6000'
29  i=i+1;f(i)='3564|sudhir Agarwal      |executive  |
    personnel  |06/07/47|7500'
30  i=i+1;f(i)='2345|j.b. saxena             |g.m.
                |marketing |12/03/45|8000'
31  i=i+1;f(i)='0110|v.k. agrawal          |g.m.
                |marketing  |31/02/40|9000'
32  n=i
33  printf("\n\n$ cat emp.lst      # to open the file
    emp.lst")
34  halt(' ')
35  u=mopen('emp.lst','wt')
36  for i=1:n
37      mfprintf(u,"%s\n",f(i))
38      printf("%s\n",f(i))
39  end
40  mclose(u)
41  halt('')
42  clc
43  li(1)='#!/bin/sh'
44  li(2)='# emp3a.sh : Using test , $0 and $# in an if-
    elif-if construct'
```

```
li(3)='#'
li(4)='if test $# -eq 0 ; then'
li(5)='      echo '+ascii(34)+'Usage: $0 pattern file'+ascii(34)+' >/dev/tty'
li(6)='elif test $# -eq 2 : then'
li(7)='     grep '+ascii(34)+'$1'+ascii(34)+' $2 || echo '+ascii(34)+'$1 not found in $2'+ascii(34)+' > /dev/tty'
li(8)='else'
li(9)='    echo '+ascii(34)+'You did not enter two arguments'+ascii(34)+' >/dev/tty'
li(10)='fi'

printf("\n# Enter the name of the shellscript file whichever you desire  \n\n")
nam=input('$ cat  ','s')
halt(' ')

for i=1:10
    printf("%s\n",li(i))
end
halt(' ')
clc

lss(1)='@echo off'
lss(2)='set x=0'
lss(3)='for %%d in (%*) do set /a x+=1'
lss(4)='if %x% equ 0 echo Usage nam pattern file&& goto endd'
lss(5)='if %x% equ 2 goto process'
lss(6)='if %x% neq 2 echo You didn'+ascii(39)+'t enter two arguments&&goto endd'
lss(7)=':process'
lss(8)='findstr '+ascii(34)+'%1'+ascii(34)+' %2> result1'
lss(9)='set b='+ascii(34)+'result1'+ascii(34)
lss(10)='for /F '+ascii(34)+'usebackq'+ascii(34)+' %%A in ('+ascii(39)+'%b%'+ascii(39)+') do set si=
```

```
    %%~zA'
74  lss(11)='if %si% equ 0 echo %1 not found in %2&&goto
      endd'
75  lss(12)='type result1'
76  lss(13)=':endd'
77  lss(14)='echo.'
78  lss(15)='pause>null'
79  lss(16)='if exist result1 del result1'
80
81
82  if getos()=='Linux' then
83      printf("\n\nPlease Switch to windows and then
          execute\n\nThank You \n\n")
84      halt(' ')
85      exit
86  end
87
88
89  v=mopen(nam+'.sh.bat','wt')
90  for i=1:16
91      mfprintf(v,"%s\n",lss(i))
92  end
93  mclose(v)
94
95
96  printf("\n# type the following command in the
      command line interpreter as soon as it appears")
97  printf(" \n        %c  %s.sh     %c [COMMANDLINE
      ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
98
99  printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
100
101 halt(' ')
102 dos('start')
103 printf("\n\n\n")
104 halt('  ------------------>Executing ShellScript in
      Command Line Prompt<---------------- ')
```

```
105 printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
106 halt(".......# (hit [ENTER] for result)")
107 //clc()
108
109 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
110 sleep(1000)
111
112 mdelete(nam+'.sh.bat')
113 mdelete('emp.lst')
```

Scilab code Exa 14.6 Program 6

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 6     :                    Checks user
     inputs for null values and runs accordingly \n")
7  disp("
   ************************************************************
   ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
    the form of a demo\n\nInitially the whole perl
    script is displaying and then \n the result of
```

```
    the same can be seen in the command line
    interpreter.\n\n2. PLEASE MAKE SURE THAT THE
    PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
    THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
    AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
    ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
    \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
16 i=0
17 i=i+1;f(i)='2233|a.k.shukla              |g.m.
          |sales          |12/12/52|6000'
18 i=i+1;f(i)='9876|jai sharma              |director   |
    production |12/03/50|7000'
19 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m      |
    marketing  |19/04/43|6000'
20 i=i+1;f(i)='2356|barun sengupta      |director   |
    personnel  |11/05/47|7800'
21 i=i+1;f(i)='5423|n.k. gupta              |chairman |
    admin      |30/08/56|5400'
22 i=i+1;f(i)='1006|chanchal singhvi    |director   |
    sales          |03/09/38|6700'
23 i=i+1;f(i)='6213|karuna ganguly      |g.m.       |
    accounts   |05/06/62|6300'
24 i=i+1;f(i)='1265|s.n. dasgupta         |manager    |
    sales          |12/09/63|5600'
25 i=i+1;f(i)='4290|jayant Choudhary  |executive  |
    production|07/09/50|6000'
26 i=i+1;f(i)='2476|anil aggarwal        |manager    |
    sales          |01/05/59|5000'
27 i=i+1;f(i)='6521|lalit chowdury       |director     |
    marketing |26/09/45|8200'
28 i=i+1;f(i)='3212|shyam saksena      |d.g.m      |
    accounts   |12/12/55|6000'
29 i=i+1;f(i)='3564|sudhir Agarwal     |executive  |
```

```
      personnel   |06/07/47|7500'
30  i=i+1;f(i)='2345|j.b. saxena              |g.m.
                  |marketing |12/03/45|8000'
31  i=i+1;f(i)='0110|v.k. agrawal             |g.m.
                  |marketing   |31/02/40|9000'
32  n=i
33  printf("\n\n$ cat emp.lst       # to open the file
      emp.lst")
34  halt(' ')
35  u=mopen('emp.lst','wt')
36  for i=1:n
37      mfprintf(u,"%s\n",f(i))
38      printf("%s\n",f(i))
39  end
40  mclose(u)
41  halt('')
42  clc
43  li(1)='#!/bin/sh'
44  li(2)='# emp4.sh : Checks user input for null values
      .Finally runs emp3a.sh'
45  li(3)='#'
46  li(4)='if  [ $# -eq 0 ]; then'
47  li(5)='       echo '+ascii(34)+'Enter the string to
      be searched : \c'+ascii(34)
48  li(6)='       read pname'
49  li(7)='  if [ -z '+ascii(34)+'$pname'+ascii(34)+' ]
      ; then'
50  li(8)='echo '+ascii(34)+'You have not entered the
      string '+ascii(34)+' ; exit 1'
51  li(9)='fi'
52  li(10)='echo '+ascii(34)+' Enter the filename to be
      used : \c '+ascii(34)
53  li(11)='read flname'
54  li(12)='if [ ! -n '+ascii(34)+'$flname'+ascii(34)+']
       ; then          #!-n same as -z'
55  li(13)='echo '+ascii(34)+'You have not entered the
      filename '+ascii(34)+' ; exit 2'
56  li(14)='fi'
```

```
57 li(15)='emp3a.sh '+ascii(34)+'$pname'+ascii(34)+'   '
     +ascii(34)+'$flname'+ascii(34)'+ascii(34)+'    #
     Runs script to do the job'
58 li(16)='elif test $# -eq 2 : then'
59 li(17)='else'
60 li(18)='emp3a.sh $*'
61 li(19)='fi'
62
63 printf("\n# Enter the name of the shellscript file
     whichever you desire  \n\n")
64 nam=input('$ cat  ','s')
65 halt(' ')
66
67 for i=1:19
68     printf("%s\n",li(i))
69 end
70 halt(' ')
71 clc
72
73 lss(1)='@echo off'
74 lss(2)='set x=0'
75 lss(3)='for %%d in (%*) do set /a x+=1'
76 lss(4)='if %x% equ 0 echo Usage nam pattern file&&
     goto endd'
77 lss(5)='if %x% equ 2 goto process'
78 lss(6)='if %x% neq 2 echo You didn'+ascii(39)+'t
     enter two arguments&&goto endd'
79 lss(7)=':process'
80 lss(8)='findstr '+ascii(34)+'%1'+ascii(34)+' %2>
     result1'
81 lss(9)='set b='+ascii(34)+'result1'+ascii(34)
82 lss(10)='for /F '+ascii(34)+'usebackq'+ascii(34)+'
     %%A in ('+ascii(39)+'%b%'+ascii(39)+') do set si=
     %%~zA'
83 lss(11)='if %si% equ 0 echo %1 not found in %2&&goto
     endd'
84 lss(12)='type result1'
85 lss(13)=':endd'
```

```
lss(14)='echo.'
lss(15)='pause>null'
lss(16)='if exist result1 del result1'

if getos()=='Linux' then
    printf("\n\nPlease Switch to windows and then
       execute\n\nThank You \n\n")
    halt(' ')
    exit
end



v=mopen('emp3a.sh.bat','wt')
for i=1:16
    mfprintf(v,"%s\n",lss(i))
end
mclose(v)

lss(1)='@echo off'
lss(2)='cls'
lss(3)='set argct=0'
lss(4)='for %%x in (%*) do set /a argct+=1'
lss(5)='if %argct% equ 0 goto intake'
lss(6)='emp3a.sh %*'
lss(7)='goto endx'
lss(8)=':intake'
lss(9)='set /p pname=Enter the string to be searched
   :'
lss(10)='if  '+ascii(34)+'%pname%'+ascii(34)+'=='+
   ascii(34)+''+ascii(34)+'  echo You have not
   entered the string&&goto endx'
lss(11)='set /p flname=Enter the filename to be used
   : '
lss(12)='if  '+ascii(34)+'%flname%'+ascii(34)+'=='+
   ascii(34)+''+ascii(34)+' echo You have not
   entered the filename&&goto endx'
lss(13)='emp3a.sh %pname% %flname%'
```

```
117 lss(14)=':endx'
118 lss(15)='pause>null'
119
120 v=mopen(nam+'.sh.bat','wt')
121 for i=1:15
122     mfprintf(v,"%s\n",lss(i))
123 end
124 mclose(v)
125
126 printf("\n# type the following command in the
      command line interpreter as soon as it appears")
127 printf(" \n        %c  %s.sh     %c [COMMANDLINE
      ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
128
129 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
130
131 halt(' ')
132 dos('start')
133 printf("\n\n\n")
134 halt('  ------------------>Executing ShellScript in
      Command Line Prompt<---------------- ')
135 printf("\n\n\n$ exit         #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
136 halt("........# (hit [ENTER] for result)")
137 //clc()
138
139 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
140 sleep(1000)
141
142 mdelete(nam+'.sh.bat')
143 mdelete('emp.lst')
144
145 mdelete('emp3a.sh.bat')
```

Scilab code Exa 14.7 Program 7

```
clear
flag=1
mode(-1)
clc

printf("Example 7     :                    Show the method
    of determining the attributes of a file in unix
   \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
   AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
   ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
   \n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
```

```
    PRELOADED COMMANDS)\n\n\n")
16  i=0
17  i=i+1;f(i)='2233|a.k.shukla              |g.m.
            |sales          |12/12/52|6000'
18  i=i+1;f(i)='9876|jai sharma              |director   |
    production |12/03/50|7000'
19  i=i+1;f(i)='5678|sumit chakrobarty|d.g.m      |
    marketing  |19/04/43|6000'
20  i=i+1;f(i)='2356|barun sengupta       |director   |
    personnel  |11/05/47|7800'
21  i=i+1;f(i)='5423|n.k. gupta              |chairman |
    admin       |30/08/56|5400'
22  i=i+1;f(i)='1006|chanchal singhvi    |director   |
    sales          |03/09/38|6700'
23  i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
    accounts   |05/06/62|6300'
24  i=i+1;f(i)='1265|s.n. dasgupta          |manager  |
    sales          |12/09/63|5600'
25  i=i+1;f(i)='4290|jayant Choudhary  |executive |
    production|07/09/50|6000'
26  i=i+1;f(i)='2476|anil aggarwal         |manager    |
    sales          |01/05/59|5000'
27  i=i+1;f(i)='6521|lalit chowdury        |director     |
    marketing |26/09/45|8200'
28  i=i+1;f(i)='3212|shyam saksena       |d.g.m      |
    accounts   |12/12/55|6000'
29  i=i+1;f(i)='3564|sudhir Agarwal      |executive  |
    personnel  |06/07/47|7500'
30  i=i+1;f(i)='2345|j.b. saxena             |g.m.
              |marketing |12/03/45|8000'
31  i=i+1;f(i)='0110|v.k. agrawal          |g.m.
              |marketing  |31/02/40|9000'
32  n=i
33  printf("\n\n$ cat emp.lst      # to open the file
    emp.lst")
34  halt(' ')
35  u=mopen('emp.lst','wt')
36  for i=1:n
```

```
37      mfprintf(u,"%s\n",f(i))
38      printf("%s\n",f(i))
39  end
40  mclose(u)
41  halt('')
42  clc
43  li(1)='#!/bin/sh'
44  li(2)='# filetest.sh : Tests file attributes'
45  li(3)='#'
46  li(4)='if [ ! -e $1 ]; then'
47  li(5)='echo '+ascii(34)+'File does not exist'+ascii
       (34)
48  li(6)='elif [! -r $1]; then '
49  li(7)='echo '+ascii(34)+'File is not readable'+ascii
       (34)
50  li(8)='elif [! -w $1]; then'
51  li(9)='echo '+ascii(34)+'File is not writable'+ascii
       (34)
52  li(10)='else'
53  li(11)='    echo '+ascii(34)+'File is both readable
       and writable'+ascii(34)
54  li(12)='    fi'
55
56
57
58  printf("\n# Enter the name of the shellscript file
       whichever you desire  \n\n")
59  nam=input('$ cat  ','s')
60  halt(' ')
61
62  for i=1:12
63      printf("%s\n",li(i))
64  end
65  halt(' ')
66  clc
67
68  lss(1)='@echo off'
69  lss(2)='if not  exist %1 echo file does not exist&&
```

```
      goto ends'
70  lss(3)='for /F "'+ascii(34)+'usebackq'+ascii(34)+'
      %%A in ('+ascii(39)+'%1'+ascii(39)+') do set
      attr=%%~aA&&set "'+ascii(34)+'wrtatt=%attr:~1,1%'+
      ascii(34)+''
71  lss(4)='if "'+ascii(34)+'%wrtatt%'+ascii(34)+'=='+
      ascii(34)+'-'+ascii(34)+' (goto rdwt) else goto
      rdnwt'
72  lss(6)=':rdwt      '
73  lss(7)='echo File is both readable and writable&&
      goto ends'
74  lss(8)=':rdnwt '
75  lss(9)='echo File is not writable'
76  lss(10)=':ends'
77  lss(11)='pause>nul'
78
79  if getos()=='Linux' then
80      printf("\n\nPlease Switch to windows and then
          execute\n\nThank You \n\n")
81      halt(' ')
82      exit
83  end
84
85
86  v=mopen(nam+'.sh.bat','wt')
87  for i=1:11
88      mfprintf(v,"%s\n",lss(i))
89  end
90  mclose(v)
91
92  if getos()=='Linux' then
93      printf("\n\nPlease open another terminal,then go
          to the directory %s  and then execute using
          the following instruction\n\n$ shell %s.sh \n
          \nThank You \n\n",curr,nam)
94      halt(' ')
95      exit
96  end
```

```
97
98
99  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
100 printf(" \n        %c  %s.sh      %c [COMMANDLINE
       ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
101 printf("\nPLEASE EXECUTE THE SCRIPT IN THE COMMAND
       PROMPT TWICE IF YOU DO NOT GET THE OUTPUT IN THE
       \n FIRST ATTEMPT.THERE IF SOME TECHNICAL ERROR\
       nSORRY FOR THE INCONVENIENCE CAUSED")
102 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                  #to execute the perlscript",nam)
103
104 halt(' ')
105 dos('start')
106 printf("\n\n\n")
107 halt('  ------------------>Executing ShellScript in
       Command Line Prompt<---------------- ')
108 printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
109 halt("........# (hit [ENTER] for result)")
110 //clc()
111
112 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
113 sleep(1000)
114
115 mdelete(nam+'.sh.bat')
116 mdelete('emp.lst')
```

Scilab code Exa 14.8 Program 8

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 8     :                    Show the method
       of using if else construct in shell progamming \
      n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
16
17 halt('')
18 clc
19 li(1)='#!/bin/sh'
20 li(2)='#menu.sh:Uses case to offer 5-item menu'
21 li(3)='#'
22 li(4)='echo '+ascii(34)+'                    MENU\n1.
```

```
      List of files\n2.Processes of user\n3. Todays
      date'
23 li(5)='4.Users of system\n5. Quit the shell\nEnter
      your option: \c'+ascii(34)+''
24 li(6)='read choice'
25 li(7)='case '+ascii(34)+'$choice'+ascii(34)+' in'
26 li(8)='    1) ls -l ;;'
27 li(9)='    2) ps -f ;;'
28 li(10)='    3)date ;;'
29 li(11)='    4)who ;;'
30 li(12)='    5)exit ;;'
31 li(13)='    *) echo '+ascii(34)+'Invalid Option'+
      ascii(34)+'         # ;; not really required for
      the last option'
32 li(14)='    end'
33 li(15)='    esac'
34
35 printf("\n# Enter the name of the shellscript file
      whichever you desire  \n\n")
36 nam=input('$ cat  ','s')
37 halt(' ')
38
39 for i=1:15
40     printf("%s\n",li(i))
41 end
42 halt(' ')
43 clc
44
45 lss(1)='@echo off'
46 lss(2)='cls'
47 lss(3)='echo.'
48 lss(4)='echo.'
49 lss(5)='echo                    MENU'
50 lss(6)=':retrn'
51 lss(7)='echo.'
52 lss(8)='echo 1. List of files'
53 lss(9)='echo 2. Processes of user'
54 lss(10)='echo 3. Today'+ascii(34)+'s date'
```

```
55 lss(11)='echo 4. Users of system'
56 lss(12)='echo 5. Quit to UNIX'
57 lss(13)='choice /c 123456 /d 6 /t 20 /n /m '+ascii
     (34)+'Enter your option'+ascii(34)+''
58 lss(14)='if ERRORLEVEL 6 pause>NUL&&echo Invalid
     option&&goto retrn'
59 lss(15)='if ERRORLEVEL 5 pause>NUL&&exit'
60 lss(16)='if ERRORLEVEL 4 pause>NUL&&net user&&goto
     ends'
61 lss(17)='if ERRORLEVEL 3 pause>NUL&&powershell date
    &&goto ends'
62 lss(18)='if ERRORLEVEL 2 pause>NUL&&tasklist&&goto
     ends'
63 lss(19)='if ERRORLEVEL 1 pause>NUL&&dir&&goto ends'
64 lss(20)=':ends'
65 lss(21)='pause>NUL'
66
67 if getos()=='Linux' then
68     printf("\n\nPlease Switch to windows and then
         execute\n\nThank You \n\n")
69     halt(' ')
70     exit
71 end
72
73
74 v=mopen(nam+'.sh.bat','wt')
75 for i=1:21
76     mfprintf(v,"%s\n",lss(i))
77 end
78 mclose(v)
79
80
81
82 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
83 printf(" \n        %c  %s.sh      %c [COMMANDLINE
     ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
84
```

```
85 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
86
87 halt(' ')
88 dos('start')
89 printf("\n\n\n")
90 halt('  ------------------>Executing ShellScript in
   Command Line Prompt<---------------- ')
91 printf("\n\n$ exit          #To exit the current
   simulation terminal and return to Scilab console\
   n\n")
92 halt(".......# (hit [ENTER] for result)")
93 //clc()
94
95 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
96 sleep(1000)
97
98 mdelete(nam+'.sh.bat')
99 mdelete('emp.lst')
```

Scilab code Exa 14.9 Program 9

```
1
2 clear
3 flag=1
4 mode(-1)
5 clc
6
7 printf("Example 9     :                Show the use of
    while loop \n")
```

```
8  disp("
      ************************************************************
      ")
9  disp("Answer     :    ")
10
11 disp("INSTRUCTIONS    : ")
12 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
17
18 halt('')
19 clc
20
21 li(1)='#!/bin/sh'
22 li(2)='# emp5.sh: Shows use of the while loop'
23 li(3)='#'
24 li(4)='answer=y'
25 li(5)='while [ '+ascii(34)+'$answer'+ascii(34)+' = '
      +ascii(34)+'y'+ascii(34)+' ]   # The control
      command'
26 li(6)='do'
27 li(7)='echo '+ascii(34)+'Enter the code and
      description: \c'+ascii(34)+'>/dev/tty'
28 li(8)='read code description # Read both together'
29 li(9)='echo '+ascii(34)+'$code|$description'+ascii
      (34)+'>>newlist  # Append a line to newlist'
```

```
30 li(10)='echo '+ascii(34)+'Enter any more(y/n)? \c'+
     ascii(34)+'>/dev/tty'
31 li(11)='read anymore'
32 li(12)='case $anymore in'
33 li(13)='      y*|Y*) answer=y ;;    # also accepts
     yes,YES etc'
34 li(14)='      n*|N*) answer=n ;;  # also accepts no,
     NO elc'
35 li(15)='           *) answer=y ;;'
36 li(16)='esac'
37 li(17)='done'
38
39 printf("\n# Enter the name of the shellscript file
     whichever you desire  \n\n")
40 nam=input('$ cat  ','s')
41 halt(' ')
42
43
44 for i=1:17
45     printf("%s\n",li(i))
46 end
47 halt(' ')
48 clc
49
50 lst(1)='@echo off&&cls'
51 lst(2)='set answer=y'
52 lst(3)=':loop'
53 lst(4)='if not '+ascii(34)+'%answer%'+ascii(34)+'=='
     +ascii(34)+'y'+ascii(34)+' goto endloop'
54 lst(5)='set /p varr=Enter the code and the
     description:  '
55 lst(6)='for /F  '+ascii(34)+'tokens=1,2* '+ascii(34)
     +' %%i in ( '+ascii(34)+'%varr%'+ascii(34)+') do
     set code=%%i&&set description=%%j '
56 lst(7)='echo %code%:%description%>>newlist'
57 lst(8)='set /p anymore=Enter any more (y/n)?  '
58 lst(9)='if '+ascii(34)+'%anymore%'+ascii(34)+'=='+
     ascii(34)+'n'+ascii(34)+' set answer=n&&goto loop
```

```
    ,
59  lst(10)='if  '+ascii(34)+'%anymore%'+ascii(34)+'=='+
      ascii(34)+'no'+ascii(34)+' set answer=n&&goto
      loop'
60  lst(11)='if  '+ascii(34)+'%anymore%'+ascii(34)+'=='+
      ascii(34)+'No'+ascii(34)+' set answer=n&&goto
      loop'
61  lst(12)='if  '+ascii(34)+'%anymore%'+ascii(34)+'=='+
      ascii(34)+'NO'+ascii(34)+' set answer=n&&goto
      loop'
62  lst(13)='if  '+ascii(34)+'%anymore%'+ascii(34)+'=='+
      ascii(34)+'N'+ascii(34)+' set answer=n&&goto loop
      '
63  lst(14)='set answer=y'
64  lst(15)='goto loop'
65  lst(16)=':endloop'
66  lst(17)='pause>NUL'
67  lst(18)='echo.&&cls'
68  lst(19)='echo Do you want to see the file newlist'
69  lst(20)='set /p chh=Enter y for Yes and n for No:  '
70  lst(21)='if  '+ascii(34)+'%chh%'+ascii(34)+'=='+
      ascii(34)+'y'+ascii(34)+' type newlist'
71  lst(22)='pause>NUL&&echo Thank you&&del newlist'
72
73
74  if getos()=='Linux' then
75      printf("\n\nPlease Switch to windows and then
          execute\n\nThank You \n\n")
76      halt(' ')
77      exit
78  end
79
80  v=mopen(nam+'.sh.bat','wt')
81  for i=1:22
82      mfprintf(v,"%s\n",lst(i))
83  end
84  mclose(v)
85
```

```
86
87
88
89 printf("\n# type the following command in the
      command line interpreter as soon as it appears")
90 printf(" \n        %c  %s.sh      %c [COMMANDLINE
      ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
91
92 printf("\n$ %s.sh   [COMMANDLINE ARGUMENTS]
                  #to execute the perlscript",nam)
93
94 halt(' ')
95 dos('start')
96 printf("\n\n\n")
97 halt('   ------------------>Executing ShellScript in
      Command Line Prompt<---------------- ')
98 printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
99 halt("........# (hit [ENTER] for result)")
100 //clc()
101
102 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
103 sleep(1000)
104
105 mdelete(nam+'.sh.bat')
```

Scilab code Exa 14.10 Program 10

```
1
```

```
2  clear
3  flag=1
4  mode(-1)
5  clc
6
7  printf("Example 10     :
                                                    Show the use
       of while loop and sleep \n")
8  disp("
      ************************************************************
      ")
9  disp("Answer     :    ")
10
11 disp("INSTRUCTIONS    : ")
12 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
17
18 halt('')
19 clc
20
21 li(1)='#! /bin/sh'
22 li(2)='# monitfile.sh:Waits for a file to be created
      '
23 li(3)='#'
24 li(4)='while [ ! -r invoice.lst ]                #
```

```
    While the file invoice.lst cannot be read'
li(5)='do '
li(6)='            sleep 60
                                # sleep for 60 seconds'
li(7)='done'
li(8)='alloc.pl
                                                # Execute
    this program after exiting the loop'

printf("\n# Enter the name of the shellscript file
    whichever you desire  \n\n")
nam=input('$ cat  ','s')
halt(' ')

for i=1:8
    printf("%s\n",li(i))
end
halt(' ')
clc

lst(1)='@echo off'
lst(2)='cls'
lst(3)='echo This program keeps on looping until the
    file invoice.lst if exists is readonly'
lst(4)='echo Later it executes the script alloc.pl
    if it exists once it if readwrite type '
lst(5)=':loop'
lst(6)='set perm=r'
lst(7)='if  '+ascii(34)+'%perm%'+ascii(34)+'='+ascii
    (34)+'-'+ascii(34)+' goto endloop'
lst(8)=' ping -n 60 localhost>null'
lst(9)=' if  exist invoice.lst for /F '+ascii(34)+'
    usebackq'+ascii(34)+' %%A in ('+ascii(39)+'
    invoice.lst'+ascii(39)+') do set att=%%~aA&&set
    perm=%att:~1,1%'
lst(10)='goto loop'
lst(11)=':endloop'
lst(12)='echo Executing alloc.pl if it exists'
```

```
52 lst(13)='    if exists alloc.pl start alloc.pl'
53 lst(14)='        pause>NUL'
54
55 if getos()=='Linux' then
56     printf("\n\nPlease Switch to windows and then
          execute\n\nThank You \n\n")
57     halt(' ')
58     exit
59 end
60
61
62 v=mopen(nam+'.sh.bat','wt')
63 for i=1:14
64     mfprintf(v,"%s\n",lst(i))
65 end
66 mclose(v)
67
68
69 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
70 printf(" \n       %c  %s.sh     %c [COMMANDLINE
     ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
71
72 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
73
74 halt(' ')
75 dos('start')
76 printf("\n\n\n")
77 halt('  ------------------>Executing ShellScript in
     Command Line Prompt<---------------- ')
78 printf("\n\n\n$ exit        #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
79 halt("........# (hit [ENTER] for result)")
80 //clc()
81
82 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
```

```
    initial environment')
83 sleep(1000)
84
85 mdelete(nam+'.sh.bat')
```

Scilab code Exa 14.11 Program 11

```
1
2 clear
3 flag=1
4 mode(-1)
5 clc
6
7 printf("Example 11     :
                                                Show the use
    of positional parameters in shells \n")
8 disp("
   ************************************************************
   ")
9 disp("Answer     :    ")
10
11 disp("INSTRUCTIONS    : ")
12 printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
   AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
   ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
```

```
    \n")
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
17 i=0
18 i=i+1;f(i)='2233|a.k.shukla                |g.m.
          |sales          |12/12/52|6000'
19 i=i+1;f(i)='9876|jai sharma                |director  |
    production |12/03/50|7000'
20 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m      |
    marketing  |19/04/43|6000'
21 i=i+1;f(i)='2356|barun sengupta      |director  |
    personnel  |11/05/47|7800'
22 i=i+1;f(i)='5423|n.k. gupta              |chairman |
    admin       |30/08/56|5400'
23 i=i+1;f(i)='1006|chanchal singhvi   |director  |
    sales          |03/09/38|6700'
24 i=i+1;f(i)='6213|karuna ganguly      |g.m.      |
    accounts   |05/06/62|6300'
25 i=i+1;f(i)='1265|s.n. dasgupta         |manager  |
    sales          |12/09/63|5600'
26 i=i+1;f(i)='4290|jayant Choudhary  |executive |
    production|07/09/50|6000'
27 i=i+1;f(i)='2476|anil aggarwal        |manager   |
    sales          |01/05/59|5000'
28 i=i+1;f(i)='6521|lalit chowdury       |director    |
    marketing |26/09/45|8200'
29 i=i+1;f(i)='3212|shyam saksena      |d.g.m      |
    accounts   |12/12/55|6000'
30 i=i+1;f(i)='3564|sudhir Agarwal     |executive  |
    personnel  |06/07/47|7500'
31 i=i+1;f(i)='2345|j.b. saxena            |g.m.
              |marketing |12/03/45|8000'
32 i=i+1;f(i)='0110|v.k. agrawal          |g.m.
              |marketing  |31/02/40|9000'
33 n=i
```

```
34 printf("\n\n$ cat emp.lst      # to open the file
      emp.lst")
35 halt(' ')
36 u=mopen('emp.lst','wt')
37 for i=1:n
38     mfprintf(u,"%s\n",f(i))
39     printf("%s\n",f(i))
40 end
41 mclose(u)
42 halt('')
43 halt('')
44 clc
45
46 li(1)='#! /bin/sh'
47 li(2)='# emp6.sh -- Using a for loop with positional
      parameters'
48 li(3)='#'
49 li(4)='for pattern in '+ascii(34)+'$@'+ascii(34)+' ;
      do              # decided not to use $* in the
      previous section'
50 li(5)='    grep '+ascii(34)+'$pattern'+ascii(34)+'
      emp.lst||echo '+ascii(34)+'pattern $pattern not
      found'+ascii(34)
51 li(6)='done'
52
53 printf("\n# Enter the name of the shellscript file
      whichever you desire  \n\n")
54 nam=input('$ cat  ','s')
55 halt(' ')
56
57 for i=1:6
58     printf("%s\n",li(i))
59 end
60 halt(' ')
61 clc
62
63 lst(1)='@echo off'
64 lst(2)='set b=0'
```

```
65 lst(3)='for %%x in (%*) do set /a b+=1'
66 lst(4)='set i=2'
67 lst(5)='findstr '+ascii(34)+'%1'+ascii(34)+' emp.lst
     >res'
68 lst(6)='for /F '+ascii(34)+'usebackq'+ascii(34)+'
     %%A in ('+ascii(39)+'res'+ascii(39)+') do set siz
     =%%~zA'
69
70 lst(7)=':loop'
71 lst(8)='if %siz% equ 0 echo Pattern %1 not found&&
     goto incr'
72 lst(9)='echo Search results for pattern %1'
73 lst(10)='echo
     ------------------------------------------------------'
74 lst(11)='echo.'
75 lst(12)='type res'
76
77 lst(13)=':incr'
78 lst(14)='if %i% gtr %b% goto endloop'
79 lst(15)='shift /1'
80 lst(16)='del res'
81 lst(17)='findstr '+ascii(34)+'%1'+ascii(34)+' emp.
     lst>res'
82 lst(18)='for /F '+ascii(34)+'usebackq'+ascii(34)+'
     %%A in ('+ascii(39)+'res'+ascii(39)+') do set siz
     =%%~zA'
83 lst(19)='set /a i+=1'
84 lst(20)='echo.'
85 lst(21)='goto loop'
86
87 lst(22)=':endloop'
88 lst(23)='pause>NUL'
89 lst(24)='del res'
90
91 if getos()=='Linux' then
92     printf("\n\nPlease Switch to windows and then
         execute\n\nThank You \n\n")
93     halt(' ')
```

```
94      exit
95  end
96
97
98  v=mopen(nam+'.sh.bat','wt')
99  for i=1:24
100     mfprintf(v,"%s\n",lst(i))
101 end
102 mclose(v)
103
104
105 printf("\n# type the following command in the
       command line interpreter as soon as it appears")
106 printf(" \n        %c  %s.sh      %c [COMMANDLINE
       ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
107
108 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
109
110 halt(' ')
111 dos('start')
112 printf("\n\n\n")
113 halt('  ------------------>Executing ShellScript in
       Command Line Prompt<---------------- ')
114 printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
115 halt("........# (hit [ENTER] for result)")
116 //clc()
117
118 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
119 sleep(1000)
120
121 mdelete(nam+'.sh.bat')
122 mdelete('emp.lst')
```

Scilab code Exa 14.12 Program 12

```

clear
flag=1
mode(-1)
clc

printf("Example 12     :                                                  Show the use
     of shift arguments \n")
disp("
    ****************************************************************
    ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
    the form of a demo\n\nInitially the whole perl
    script is displaying and then \n the result of
    the same can be seen in the command line
    interpreter.\n\n2. PLEASE MAKE SURE THAT THE
    PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
    THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
    AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
    ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
    \n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
```

```
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
17 i=0
18 i=i+1;f(i)='2233|a.k.shukla                |g.m.
            |sales            |12/12/52|6000'
19 i=i+1;f(i)='9876|jai sharma                |director   |
     production |12/03/50|7000'
20 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m       |
     marketing  |19/04/43|6000'
21 i=i+1;f(i)='2356|barun sengupta       |director   |
     personnel  |11/05/47|7800'
22 i=i+1;f(i)='5423|n.k. gupta               |chairman |
     admin       |30/08/56|5400'
23 i=i+1;f(i)='1006|chanchal singhvi    |director   |
     sales          |03/09/38|6700'
24 i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
     accounts   |05/06/62|6300'
25 i=i+1;f(i)='1265|s.n. dasgupta          |manager  |
     sales          |12/09/63|5600'
26 i=i+1;f(i)='4290|jayant Choudhary  |executive |
     production|07/09/50|6000'
27 i=i+1;f(i)='2476|anil aggarwal         |manager    |
     sales          |01/05/59|5000'
28 i=i+1;f(i)='6521|lalit chowdury        |director     |
     marketing |26/09/45|8200'
29 i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
     accounts   |12/12/55|6000'
30 i=i+1;f(i)='3564|sudhir Agarwal      |executive  |
     personnel  |06/07/47|7500'
31 i=i+1;f(i)='2345|j.b. saxena              |g.m.
               |marketing |12/03/45|8000'
32 i=i+1;f(i)='0110|v.k. agrawal          |g.m.
               |marketing  |31/02/40|9000'
33 n=i
34 printf("\n\n$ cat emp.lst      # to open the file
     emp.lst")
35 halt(' ')
36 u=mopen('emp.lst','wt')
```

```
37 for i=1:n
38     mfprintf(u,"%s\n",f(i))
39     printf("%s\n",f(i))
40 end
41 mclose(u)
42 halt('')
43 halt('')
44 clc
45
46 li(1)='#!/bin/sh'
47 li(2)='# emp7.sh: Script using shift -- Saves first
     argument;for works with the rest'
48 li(3)='#'
49 li(4)='case $# in'
50 li(5)='    0|1) echo '+ascii(34)+'Usage: $0 file
     pattern(s)'+ascii(34)+' ; exit 2 ;;'
51 li(6)='    *) flname=$1        # store $1 as a
     variable before it gets lost'
52 li(7)='       shift'
53 li(8)='       for pattern in '+ascii(34)+'$@'+ascii
     (34)+' ; do     # Starts iteration with $2'
54 li(9)='                grep '+ascii(34)+'$pattern'+
     ascii(34)+' $flname || echo '+ascii(34)+'Pattern
     $pattern not found'+ascii(34)
55 li(10)='       done'
56 li(11)='  esac'
57
58
59 printf("\n# Enter the name of the shellscript file
     whichever you desire  \n\n")
60 nam=input('$ cat  ','s')
61 halt(' ')
62
63 for i=1:11
64     printf("%s\n",li(i))
65 end
66 halt(' ')
67 clc
```

```
68
69
70 lst(1)='@echo off'
71 lst(2)='set b=0'
72 lst(3)='for %%x in (%*) do set /a b+=1'
73 lst(4)='set i=3'
74 lst(5)='set fille=%1'
75 lst(6)='shift /1'
76 lst(7)='findstr '+ascii(34)+'%1'+ascii(34)+' %fille%
     >res'
77 lst(8)='for /F '+ascii(34)+'usebackq'+ascii(34)+'
     %%A in ('+ascii(39)+'res'+ascii(39)+') do set siz
     =%%~zA'
78 lst(9)=':loop'
79 lst(10)='if %siz% equ 0 echo Pattern %1 not found&&
     goto incr'
80 lst(11)='echo Search results for pattern %1'
81 lst(12)='echo
     ------------------------------------------------',
82 lst(13)='echo.'
83 lst(14)='type res'
84 lst(15)=':incr'
85 lst(16)='if %i% gtr %b% goto endloop'
86 lst(17)='shift /1'
87 lst(18)='del res'
88 lst(19)='findstr '+ascii(34)+'%1'+ascii(34)+'
     %fille%>res'
89 lst(20)='for /F '+ascii(34)+'usebackq'+ascii(34)+'
     %%A in ('+ascii(39)+'res'+ascii(39)+') do set siz
     =%%~zA'
90 lst(21)='set /a i+=1'
91 lst(22)='echo.'
92 lst(23)='goto loop'
93 lst(24)=':endloop'
94 lst(25)='pause>NUL'
95 lst(26)='del res'
96
97
```

```
98  if getos()=='Linux' then
99      printf("\n\nPlease Switch to windows and then
            execute\n\nThank You \n\n")
100     halt(' ')
101     exit
102 end
103
104
105 v=mopen(nam+'.sh.bat','wt')
106 for i=1:26
107     mfprintf(v,"%s\n",lst(i))
108 end
109 mclose(v)
110
111
112 printf("\n# type the following command in the
       command line interpreter as soon as it appears")
113 printf(" \n        %c  %s.sh      %c [COMMANDLINE
       ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
114
115 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                  #to execute the perlscript",nam)
116
117 halt(' ')
118 dos('start')
119 printf("\n\n\n")
120 halt('  ------------------>Executing ShellScript in
       Command Line Prompt<---------------- ')
121 printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
122 halt("........# (hit [ENTER] for result)")
123 //clc()
124
125 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
126 sleep(1000)
127
```

```
128 mdelete(nam+'.sh.bat')
129 mdelete('emp.lst')
```

Scilab code Exa 14.13 Program 14

```
1
2 clear
3 flag=1
4 mode(-1)
5 clc
6
7 printf("Example 12     :
                                                Show the use
    of set command and the here documenting \n")
8 disp("
   ************************************************************
   ")
9 disp("Answer     :    ")
10
11 disp("INSTRUCTIONS    : ")
12 printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
   AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
   ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
   \n")
13 halt('.............Press [ENTER] to continue.....')
```

```
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
17 i=0
18 i=i+1;f(i)='01|accounts|6213'
19 i=i+1;f(i)='02|admin|5423'
20 i=i+1;f(i)='03|marketing|6521'
21 i=i+1;f(i)='04|personnel|2365'
22 i=i+1;f(i)='05|production|9876'
23 i=i+1;f(i)='06|sales|1006'
24 n=i
25 printf("\n\n$ cat limitlist      # to open the file
     emp.lst")
26 halt(' ')
27 u=mopen('limitlist','wt')
28 for i=1:n
29     mfprintf(u,"%s\n",f(i))
30     printf("%s\n",f(i))
31 end
32 mclose(u)
33 halt('')
34 halt('')
35 clc
36
37 i=0
38 i=i+1;f(i)='#! /bin/sh'
39 i=i+1;f(i)='# valcode.sh : Uses a here document to
     look up for a code list'
40 i=i+1;f(i)='#'
41 i=i+1;f(i)='IFS='+ascii(34)+'|'+ascii(34)+'     #
     Reset field seperator'
42 i=i+1;f(i)='while echo '+ascii(34)+'Enter department
     code  : \c'+ascii(34)+':do'
43 i=i+1;f(i)='read dcode'
44 i=i+1;f(i)='set -- `grep '+ascii(34)+'^$dcode'+ascii
     (34)+' << limit'
45 i=i+1;f(i)='01|accounts|6213'
```

```
46 i=i+1;f(i)='02|admin|5423'
47 i=i+1;f(i)='03|marketing|6521'
48 i=i+1;f(i)='04|personnel|2365'
49 i=i+1;f(i)='05|production|69876'
50 i=i+1;f(i)='06|sales|1006'
51 i=i+1;f(i)='limit``'
52 i=i+1;f(i)='              # Closing ` marks end of
     standard input'
53 i=i+1;f(i)='   case $# in'
54 i=i+1;f(i)='             3) echo '+ascii(34)+'
     Department name      : $2\nEmp-id of head of dept
     : $3\n'+ascii(34)
55 i=i+1;f(i)='               shift 3;;    #Flush out
     the positional parameters'
56 i=i+1;f(i)='             *) echo '+ascii(34)+'Invalid
      code'+ascii(34)+'  ; continue'
57 i=i+1;f(i)='esac'
58 i=i+1;f(i)='done'
59 n=i
60
61 printf("\n# Enter the name of the shellscript file
     whichever you desire  \n\n")
62 nam=input('$ cat  ','s')
63 halt(' ')
64
65 for i=1:n
66     printf("%s\n",f(i))
67 end
68 halt(' ')
69 clc
70
71
72 i=0
73 i=i+1;f(i)='@echo off'
74 i=i+1;f(i)='echo Executing Validation Programme'
75 i=i+1;f(i)='set Continue?...'
76 i=i+1;f(i)='pause>NUL'
77 i=i+1;f(i)='set chh=y'
```

```
78  i=i+1;f(i)=':loop'
79  i=i+1;f(i)='if /I not '+ascii(34)+'%chh%'+ascii(34)+
       '=='+ascii(34)+'y'+ascii(34)+' goto endloop '
80  i=i+1;f(i)='set /P dcode=Enter department code  :  '
81  i=i+1;f(i)='del res'
82  i=i+1;f(i)='del lst'
83  i=i+1;f(i)='findstr /B '+ascii(34)+'%dcode%'+ascii
       (34)+' limitlist.txt>res'
84  i=i+1;f(i)='for /F '+ascii(34)+'usebackq'+ascii(34)+
       ' %%A in ('+ascii(39)+'res'+ascii(39)+') do set
       siz=%%~zA'
85  i=i+1;f(i)='if %siz% equ 0 echo Invalid code&&set
       chh=y&&goto loop'
86
87  i=i+1;f(i)='for /F '+ascii(34)+'tokens=2,3 delims=|'
       +ascii(34)+' %%i in (res) do set dname=%%i&&set
       id=%%j'
88  i=i+1;f(i)='echo Department name          : %dname%'
89  i=i+1;f(i)='echo Emp-id of head of dept   : %id%'
90
91
92  i=i+1;f(i)='echo.'
93  i=i+1;f(i)='set /P chh=Continue?(y/n)  :'
94  i=i+1;f(i)='goto loop'
95  i=i+1;f(i)=':endloop'
96  i=i+1;f(i)='pause>NUL'
97  n=i
98
99  if getos()=='Linux' then
100     printf("\n\nPlease Switch to windows and then
           execute\n\nThank You \n\n")
101     halt(' ')
102     exit
103 end
104
105
106 v=mopen(nam+'.sh.bat','wt')
107 for i=1:n
```

```
108     mfprintf(v,"%s\n",f(i))
109 end
110 mclose(v)
111
112
113 printf("\n# type the following command in the
    command line interpreter as soon as it appears")
114 printf(" \n        %c  %s.sh      %c [COMMANDLINE
    ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
115
116 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
117
118 halt(' ')
119 dos('start')
120 printf("\n\n\n")
121 halt('  ------------------>Executing ShellScript in
    Command Line Prompt<---------------- ')
122 printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
123 halt("........# (hit [ENTER] for result)")
124 //clc()
125
126 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
127 sleep(1000)
128
129 mdelete(nam+'.sh.bat')
130 mdelete('limitlist')
```

Chapter 18

awk An Advanced Filter

Scilab code Exa 18.1 Program 1

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 1     :                  Show the method
       of using field extraction in awk \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
```

```
    ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
    \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
16 i=0
17 i=i+1;f(i)='2233|a.k.shukla              |g.m.
          |sales          |12/12/52|6000'
18 i=i+1;f(i)='9876|jai sharma              |director  |
    production |12/03/50|7000'
19 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m      |
    marketing  |19/04/43|6000'
20 i=i+1;f(i)='2356|barun sengupta      |director  |
    personnel  |11/05/47|7800'
21 i=i+1;f(i)='5423|n.k. gupta              |chairman |
    admin      |30/08/56|5400'
22 i=i+1;f(i)='1006|chanchal singhvi   |director  |
    sales          |03/09/38|6700'
23 i=i+1;f(i)='6213|karuna ganguly      |g.m.      |
    accounts   |05/06/62|6300'
24 i=i+1;f(i)='1265|s.n. dasgupta         |manager  |
    sales          |12/09/63|5600'
25 i=i+1;f(i)='4290|jayant Choudhary  |executive |
    production|07/09/50|6000'
26 i=i+1;f(i)='2476|anil aggarwal        |manager   |
    sales          |01/05/59|5000'
27 i=i+1;f(i)='6521|lalit chowdury       |director    |
    marketing |26/09/45|8200'
28 i=i+1;f(i)='3212|shyam saksena      |d.g.m      |
    accounts   |12/12/55|6000'
29 i=i+1;f(i)='3564|sudhir Agarwal     |executive  |
    personnel  |06/07/47|7500'
30 i=i+1;f(i)='2345|j.b. saxena            |g.m.
          |marketing |12/03/45|8000'
31 i=i+1;f(i)='0110|v.k. agrawal          |g.m.
          |marketing  |31/02/40|9000'
```

```
32 n=i
33 printf("\n\n$ cat emp.lst       # to open the file
      emp.lst")
34 halt(' ')
35 u=mopen('empn.lst','wt')
36 for i=1:n
37     mfprintf(u,"%s\n",f(i))
38     printf("%s\n",f(i))
39 end
40 mclose(u)
41 halt('')
42 clc
43
44 i=0
45 i=i+1;f(i)='BEGIN {IFS='+ascii(34)+'|'+ascii(34)
46 i=i+1;f(i)='printf '+ascii(34)+'\t\tEmployee
      abstract\n\n'+ascii(34)
47 i=i+1;f(i)='} $6 > 7500 {             # Increemnt the
      variables for serial number and pay'
48 i=i+1;f(i)='  kount++ ; tot+= $6       # Multiple
      assignments in one line'
49 i=i+1;f(i)='  printf '+ascii(34)+'%3d  % -20s % -12s
      %d\n'+ascii(34)+',kount,$2,$3,$6'
50 i=i+1;f(i)='}'
51 i=i+1;f(i)='END {'
52 i=i+1;f(i)='     printf '+ascii(34)+'\n\tThe averge
      basic pay is %6d\n'+ascii(34)+',tot/kount'
53 i=i+1;f(i)='}'
54 n=i
55
56 printf("\n# Enter the name of the shellscript file
      whichever you desire  \n\n")
57 nam=input('$ cat  ','s')
58 halt(' ')
59
60 for i=1:n
61     printf("%s\n",f(i))
62 end
```

```
63 halt(' ')
64 clc
65 i=0
66 i=i+1;f(i)='@echo off'
67 i=i+1;f(i)='cls'
68 i=i+1;f(i)='echo  Employee abstract'
69 i=i+1;f(i)='echo.'
70 i=i+1;f(i)='set t=0'
71 i=i+1;f(i)='set tot=0'
72 i=i+1;f(i)='for /F '+ascii(34)+'tokens=2,3,6 delims
      =|'+ascii(34)+' %%i in (%1) do if %%k gtr 7500
      set /a t+=1&&echo %%i  %%j  %%k>>res&&set /a tot
      +=%%k'
73 i=i+1;f(i)='type res'
74 i=i+1;f(i)='echo.'
75 i=i+1;f(i)='set /a tot/=i'
76 i=i+1;f(i)='echo The average basic pay is %tot%'
77 i=i+1;f(i)='del res'
78 n=i
79
80
81 if getos()=='Linux' then
82     printf("\n\nPlease Switch to windows and then
         execute using the instructions\n\nThank You \
         n\n")
83     halt(' ')
84     exit
85 end
86
87 v=mopen(nam+'.awk.bat','wt')
88 for i=1:n
89     mfprintf(v,"%s\n",f(i))
90 end
91 mclose(v)
92
93
94 printf("\n# type the following command in the
      command line interpreter as soon as it appears")
```

```
95 printf(" \n        %c    %s.awk empn.lst      %c [ENTER
   ]\n\n",ascii(34),nam,ascii(34))
96
97 printf("\n$ %s.awk    empn.lst             #to execute
   the perlscript",nam)
98
99 halt(' ')
100 dos('start')
101 printf("\n\n\n")
102 halt('   ------------------>Executing awkScript in
   Command Line Prompt<---------------- ')
103 printf("\n\n\n$ exit          #To exit the current
   simulation terminal and return to Scilab console\
   n\n")
104 halt("........# (hit [ENTER] for result)")
105 //clc()
106
107 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
108 sleep(1000)
109
110 mdelete(nam+'.awk.bat')
111 mdelete('empn.lst')
```

Scilab code Exa 18.2 Program 2

```
1 clear
2 flag=1
3 mode(-1)
4 clc
5
```

```
6  printf("Example 2    :                Show the method
       of using field extraction and begin-end in awk \
     n")
7  disp("
     ************************************************************
     ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
     the form of a demo\n\nInitially the whole perl
     script is displaying and then \n the result of
     the same can be seen in the command line
     interpreter.\n\n2. PLEASE MAKE SURE THAT THE
     PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
     THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
     AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
     ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
     \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
16 i=0
17 i=i+1;f(i)='2233|a.k.shukla              |g.m.
          |sales          |12/12/52|6000'
18 i=i+1;f(i)='9876|jai sharma              |director   |
     production |12/03/50|7000'
19 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m      |
     marketing  |19/04/43|6000'
20 i=i+1;f(i)='2356|barun sengupta       |director   |
     personnel  |11/05/47|7800'
21 i=i+1;f(i)='5423|n.k. gupta              |chairman |
     admin       |30/08/56|5400'
22 i=i+1;f(i)='1006|chanchal singhvi    |director   |
     sales          |03/09/38|6700'
23 i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
```

```
    accounts    |05/06/62|6300'
24  i=i+1;f(i)='1265|s.n. dasgupta          |manager   |
    sales           |12/09/63|5600'
25  i=i+1;f(i)='4290|jayant Choudhary   |executive |
    production|07/09/50|6000'
26  i=i+1;f(i)='2476|anil aggarwal          |manager    |
    sales           |01/05/59|5000'
27  i=i+1;f(i)='6521|lalit chowdury        |director     |
    marketing |26/09/45|8200'
28  i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
    accounts    |12/12/55|6000'
29  i=i+1;f(i)='3564|sudhir Agarwal      |executive   |
    personnel   |06/07/47|7500'
30  i=i+1;f(i)='2345|j.b. saxena              |g.m.
                |marketing |12/03/45|8000'
31  i=i+1;f(i)='0110|v.k. agrawal           |g.m.
                |marketing   |31/02/40|9000'
32  n=i
33  printf("\n\n$ cat emp.lst       # to open the file
    emp.lst")
34  halt(' ')
35  u=mopen('empn.lst','wt')
36  for i=1:n
37      mfprintf(u,"%s\n",f(i))
38      printf("%s\n",f(i))
39  end
40  mclose(u)
41  halt('')
42  clc
43
44  i=0
45  i=i+1;f(i)='begin {'
46  i=i+1;f(i)='fs = '+ascii(34)+'|'+ascii(34)+''
47  i=i+1;f(i)='printf '+ascii(34)+'%46s\n'+ascii(34)+',
      '+ascii(34)+'Basic       Da       Hra    Gross'+
    ascii(34)+''
48  i=i+1;f(i)='}/sales|marketing/  {'
49  i=i+1;f(i)='     # Calculate the da, hra and gross
```

```
    pay'
50  i=i+1;f(i)='      da = 0.25*$6 ; hra = 0.50*$6 ; gp =
     $6+hra+da'
51
52  i=i+1;f(i)='      # Store the aggregates in seperate
    arrays'
53  i=i+1;f(i)='      tot[1] += $6 ; tot[6] += da ; tot
    [3] += hra ; tot[4] += gp'
54  i=i+1;f(i)='      kount++'
55  i=i+1;f(i)='}'
56  i=i+1;f(i)='END { # Print the averages'
57  i=i+1;f(i)='     printf '+ascii(34)+'\t
    Average  %5d %5d %5d %5d\n'+ascii(34)+' , \'
58  i=i+1;f(i)='      tot[1]/kount, tot[2]/kount, tot[3]/
    kount, tot[4]/kount'
59  i=i+1;f(i)='}'
60  n=i
61
62  printf("\n# Enter the name of the shellscript file
    whichever you desire  \n\n")
63  nam=input('$ cat  ','s')
64  halt(' ')
65
66  for i=1:n
67      printf("%s\n",f(i))
68  end
69  halt(' ')
70  clc
71
72
73
74  i=0
75  i=i+1;f(i)='@echo off'
76  i=i+1;f(i)='cls'
77  i=i+1;f(i)='if exist temp.lst del temp.lst'
78  i=i+1;f(i)='if exist cntl del cntl'
79  i=i+1;f(i)='findstr /N /R '+ascii(34)+'^'+ascii(34)+
    ' %1>temp.lst'
```

```
80  i=i+1;f(i)='findstr /N /R '+ascii(34)+'^'+ascii(34)+
       ' %1|find /C '+ascii(34)+':'+ascii(34)+'>cntl'
81  i=i+1;f(i)='for /F '+ascii(34)+'delims=  '+ascii(34)
       +' %%i in (cntl) do set max=%%i'
82  i=i+1;f(i)='del cntl'
83  i=i+1;f(i)='set kount=1'
84  i=i+1;f(i)='if exist lin del lin'
85  i=i+1;f(i)='set tot1=0'
86  i=i+1;f(i)='set tot2=0'
87  i=i+1;f(i)='set tot3=0'
88  i=i+1;f(i)='set tot4=0'
89  i=i+1;f(i)=':loop'
90  i=i+1;f(i)='if %kount% gtr %max% goto endloop'
91  i=i+1;f(i)='findstr /B '+ascii(34)+'%kount%'+ascii
       (34)+' temp.lst>lin'
92  i=i+1;f(i)='for /F '+ascii(34)+'tokens=6 delims=|'+
       ascii(34)+' %%i in (lin) do set basic=%%i'
93  i=i+1;f(i)='set /a da=basic/4'
94  i=i+1;f(i)='set /a hra=basic/2'
95  i=i+1;f(i)='set /a gp=basic+da+hra'
96  i=i+1;f(i)='set /a tot1+=basic&&set /a tot2+=da&&set
       /a tot3+=hra&&set /a tot4+=gp'
97  i=i+1;f(i)='set /a kount+=1'
98  i=i+1;f(i)='goto loop'
99  i=i+1;f(i)=':endloop'
100 i=i+1;f(i)='set /a '+ascii(39)+'tot1/=kount,tot2/=
       kount,tot3/=kount,tot4/=kount'+ascii(39)+''
101 i=i+1;f(i)='echo      Average  %tot1%  %tot2%
       %tot3%  %tot4%'
102 i=i+1;f(i)='pause>NUL'
103 n=i
104
105 v=mopen(nam+'.awk.bat','wt')
106 for i=1:n
107     mfprintf(v,"%s\n",f(i))
108 end
109 mclose(v)
110
```

```
111 if getos()=='Linux' then
112     printf("\n\nPlease Switch to windows and then
        execute using the instructions\n\nThank You \
        n\n")
113     halt(' ')
114     exit
115 end
116 printf("\n# type the following command in the
    command line interpreter as soon as it appears")
117 printf(" \n       %c   %s.awk empn.lst      %c [ENTER
    ]\n\n",ascii(34),nam,ascii(34))
118
119 printf("\n$ %s.awk   empn.lst           #to execute
    the perlscript",nam)
120
121 halt(' ')
122 dos('start')
123 printf("\n\n\n")
124 halt('   ------------------>Executing awkScript in
    Command Line Prompt<---------------- ')
125 printf("\n\n\n$ exit         #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
126 halt("........# (hit [ENTER] for result)")
127 //clc()
128
129 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
130 sleep(1000)
131
132 mdelete(nam+'.awk.bat')
133 mdelete('empn.lst')
```

Chapter 19

perl The Master Manipulator

Scilab code Exa 19.1 Program 1

```
1  clear
2  pwd
3  curr=ans
4  flag=1
5  mode(-1)
6  clc
7
8
9  disp("INSTRUCTIONS    : ")
10 printf("\nHere all instructions are preloaded in the
       form of a demo\n\nInitially the whole perl
     script is displaying and then \n the result of
     the same can be seen in the command line
     interpreter.\nPLEASE MAKE SURE THAT THE
     PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
     THE COMMAND WOULD NOT WORK \n\n\nPRESS ENTER
     AFTER EACH COMMAND to see its RESULT\nPRESS ENTER
       AFTER EACH RESULT TO GO TO THE NEXT COMMAND\n")
11 halt('.............Press [ENTER] to continue.....')
12 halt("")
13 clc
```

```
14 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
15
16
17 printf("\n# Enter the name of the perlscript file
     whichever you desire  \n\n")
18 nam=input('$ cat  ','s')
19 halt(' ')
20 clc
21 li(1)='#!/usr/bin/perl'
22 li(2)='# Script: '+nam+'.pl - Shows the use of
     variables'
23 li(3)="#"
24 li(4)='print('+ascii(34)+'Enter your name:   '+ascii
     (34)+')  ;'
25 li(5)='$name = <STDIN> ;
                                          #Input from
     the keyboard'
26 li(6)='print('+ascii(34)+'Enter a temperature in
     Centigrade:  '+ascii(34)+')  ;'
27 li(7)='$centigrade=<STDIN>;
                                      #Whitespace
     unimportant'
28 li(8)='$fahrenheit=$centigrade*9/5 + 32 ;
                          #Here too '
29 li(9)='print '+ascii(34)+'The temperature $name in
     Fahrenheit is $fahrenheit\n'+ascii(34)+'  ;'
30 li(10)='print('+ascii(34)+'\n\nType exit to go back
     to console\n\n'+ascii(34)+')'
31 halt(' ')
32
33 v=mopen(nam+'.pl','wt')
34 for i=1:10
35     mfprintf(v,"%s\n",li(i))
36     if i~=10 then
37     printf("%s\n",li(i))
38     end
39 end
```

```
40 mclose(v)
41
42 if getos()=='Linux' then
43     printf("\n\nPlease open a new terminal window
          and then go to the directory %s and execute
          the following instruction\n\nperl %s.pl [
          Command line parameters if any]\n\nThank You
          \n\n",curr,nam)
44     halt(' ')
45     exit
46 end
47
48 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
49 printf(" \n       %c perl %s.pl  %c[ENTER]\n\n",
     ascii(34),nam,ascii(34))
50
51 printf("\n$ perl %s.pl    #to execute the perlscript
     ",nam)
52 halt(' ')
53 dos('start')
54 printf("\n\n\n")
55 halt('   ------------------>Executing PerlScript in
     Command Line Prompt<---------------- ')
56 printf("\n\n\n$ exit         #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
57 halt("........# (hit [ENTER] for result)")
58 //clc()
59
60 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
61 sleep(1000)
62
63 mdelete(nam+'.pl')
```

Scilab code Exa 19.2 Program 2

```
clear
flag=1
mode(-1)
pwd
curr=ans
clc

printf("Example 2     :                    Show the method
    of using chop in perl \n\n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\nHere all instructions are preloaded in the
    form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\nPLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n\nPRESS ENTER
   AFTER EACH COMMAND to see its RESULT\nPRESS ENTER
    AFTER EACH RESULT TO GO TO THE NEXT COMMAND\n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
```

```
    PRELOADED COMMANDS)\n\n\n")
18
19
20  printf("\n# Enter the name of the perlscript file
       whichever you desire  \n\n")
21  nam=input('$ cat  ','s')
22  halt(' ')
23  clc
24  li(1)='#!/usr/bin/perl'
25  li(2)='# Script: '+nam+'.pl - Demonstrates use of
       chop'
26  li(3)="#"
27  li(4)='print('+ascii(34)+'Enter your name:   '+ascii
       (34)+')  ;'
28  li(5)='$name = <STDIN> ;                '
29  li(6)='chop($name)  ;
                                                    #
       Removes newline character from $name'
30  li(7)='if ( $name ne '+ascii(34)+ascii(34)+' ) {'+
       ascii(10)+'print('+ascii(34)+'$name, have a nice
       day\n'+ascii(34)+');} '
31  li(8)='else {'+ascii(10)+'print('+ascii(34)+'You
       have not entered your name\n'+ascii(34)+');} '
32  li(9)='print('+ascii(34)+'\n\nType exit to go back
       to console\n\n'+ascii(34)+')'
33  halt(' ')
34
35  v=mopen(nam+'.pl','wt')
36  for i=1:9
37      mfprintf(v,"%s\n",li(i))
38      if i~=9 then
39      printf("%s\n",li(i))
40      end
41  end
42  mclose(v)
43  if getos()=='Linux' then
44      printf("\n\nPlease open a new terminal window
           and then go to the directory %s and execute
```

```
        the following instruction\n\nperl %s.pl [
        Command line parameters if any]\n\nThank You
        \n\n",curr,nam)
45      halt(' ')
46      exit
47  end
48
49  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
50  printf(" \n        %c perl %s.pl  %c[ENTER]\n\n",
       ascii(34),nam,ascii(34))
51
52  printf("\n$ perl %s.pl     #to execute the perlscript
       ",nam)
53  halt(' ')
54  dos('start')
55  printf("\n\n\n")
56  halt('   ------------------>Executing PerlScript in
       Command Line Prompt<---------------- ')
57  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
58  halt("........# (hit [ENTER] for result)")
59  //clc()
60
61  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
        initial environment')
62  sleep(1000)
63
64  mdelete(nam+'.pl')
```

Scilab code Exa 19.3 Program 3

```
clear
flag=1
mode(-1)
pwd
curr=ans
clc

printf("Example 3     :                    Show the method
    of using default variables in perl using \n\t\
   tmailbox specific extraction as an example \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. THIS IS
   POSSIBLE ONLY IF THE HOME DIRECTORY CONTAINS
   MAILBOX PATH\n4. PRESS ENTER AFTER EACH COMMAND
   to see its RESULT\n5. PRESS ENTER AFTER EACH
   RESULT TO GO TO THE NEXT COMMAND\n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
   PRELOADED COMMANDS)\n\n\n")


printf("\n# Enter the name of the perlscript file
   whichever you desire  \n\n")
```

```
21 nam=input('$ cat  ','s')
22 halt(' ')
23 clc
24 li(1)='#!/usr/bin/perl'
25 li(2)='# Script: '+nam+'.pl - Extracts the From:
     headers from the mailbox'
26 li(3)="#"
27 li(4)='while(<>) {

     # Actually ($_ = <>) '
28 li(5)='          chop()    ;
                                                  #
     chop($_)'
29 li(6)='          if  ( /From:.*\@velvet.com/)  {
              # if ($_ =~ /From:.*\@velvet ...) '
30 li(7)='          $slno++  ;'
31 li(8)='           print($slno  .  '+ascii(34)+'  '+
     ascii(34)+' . $_  .  '+ascii(34)+'\n'+ascii(34)+'
     )   ;'
32 li(9)='  }  } '
33 li(10)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
     '+ascii(39)+'to go back to console\n\n'+ascii(34)
     +')'
34 halt(' ')
35
36 v=mopen(nam+'.pl','wt')
37 for i=1:10
38     mfprintf(v,"%s\n",li(i))
39     if i~=10 then
40     printf("%s\n",li(i))
41     end
42 end
43 mclose(v)
44 if getos()=='Linux' then
45     printf("\n\nPlease open a new terminal window
         and then go to the directory %s and execute
         the following instruction\n\nperl %s.pl [
         Command line parameters if any]\n\nThank You
```

```
        \n\n",curr,nam)
46      halt(' ')
47      exit
48  end
49
50
51  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
52  printf(" \n        %c perl %s.pl     [MAILBOX PATH AS
       COMMANDLINE ARGUMENT] %c[ENTER]\n\n",ascii(34),
       nam,ascii(34))
53
54  printf("\n$ perl %s.pl  [MAILBOX PATH-CMDLINE
       ARGUMENT]          #to execute the perlscript",
       nam)
55  halt(' ')
56  dos('start')
57  printf("\n\n\n")
58  halt('  ------------------>Executing PerlScript in
       Command Line Prompt<---------------- ')
59  printf("\n\n\n$ exit         #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
60  halt("........# (hit [ENTER] for result)")
61  //clc()
62
63  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
        initial environment')
64  sleep(1000)
65
66  mdelete(nam+'.pl')
```

Scilab code Exa 19.4 Program 4

```
1  clear
2  flag=1
3  mode(-1)
4  pwd
5  curr=ans
6  clc
7
8  printf("Example 4     :                    Show the method
       of array handling in perl \n")
9  disp("
      ************************************************************
      ")
10 disp("Answer     :    ")
11
12 disp("INSTRUCTIONS    : ")
13 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
14 halt('.............Press [ENTER] to continue.....')
15 halt("")
16 clc
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
18
19
20 printf("\n# Enter the name of the perlscript file
      whichever you desire  \n\n")
21 nam=input('$ cat  ','s')
22 halt(' ')
```

```
23 clc
24 li(1)='#!/usr/bin/perl'
25 li(2)='# Script: '+nam+'.pl - Shows use of arrays'
26 li(3)="#"
27 li(4)='@days_between = ('+ascii(34)+'Wed'+ascii(34)+
     ', '+ascii(34)+'Thu'+ascii(34)+') ;'
28 li(5)='@days = (Mon,Tue,@days_between,Fri) ; '
29 li(6)='@days[5,6] = qw/Sat Sun/ ; '
30 li(7)='$length = @days ; '
31 li(8)='#'
32 li(9)='print('+ascii(34)+'The third day of the week
     is $days[2]\n'+ascii(34)+') ;'
33 li(10)='print('+ascii(34)+'The days of the week are
     @days\n'+ascii(34)+') ;'
34 li(11)='print('+ascii(34)+'The number of elements in
     the array is $length\n'+ascii(34)+') ;'
35 li(12)='print('+ascii(34)+'The last subscript of the
     array is $#days\n'+ascii(34)+') ;'
36 li(13)='$#days = 5;

     #Resize the array'
37 li(14)='print('+ascii(34)+'\$days[6] is now $days
     [6]\n'+ascii(34)+') ;'
38 li(15)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
     '+ascii(39)+'to go back to console\n\n'+ascii(34)
     +')'
39 halt(' ')
40
41 v=mopen(nam+'.pl','wt')
42 for i=1:15
43     mfprintf(v,"%s\n",li(i))
44     if i~=15 then
45     printf("%s\n",li(i))
46     end
47 end
48 mclose(v)
49 if getos()=='Linux' then
50     printf("\n\nPlease open a new terminal window
```

```
        and then go to the directory %s and execute
        the following instruction\n\nperl %s.pl [
        Command line parameters if any]\n\nThank You
        \n\n",curr,nam)
51      halt(' ')
52      exit
53  end
54
55
56  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
57  printf(" \n       %c perl %s.pl     %c[ENTER]\n\n",
       ascii(34),nam,ascii(34))
58
59  printf("\n$ perl %s.pl                #to execute the
        perlscript",nam)
60  halt(' ')
61  dos('start')
62  printf("\n\n\n")
63  halt('   ------------------>Executing PerlScript in
       Command Line Prompt<------------------ ')
64  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
65  halt("........# (hit [ENTER] for result)")
66  //clc()
67
68  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
        initial environment')
69  sleep(1000)
70
71  mdelete(nam+'.pl')
```

Scilab code Exa 19.5 Program 5

```
clear
flag=1
mode(-1)
pwd
curr=ans
clc

printf("Example 5     :                    Show the method
    of command line argument  handling in perl \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
   AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
   ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
   \n")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
```

```
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
18
19
20 printf("\n# Enter the name of the perlscript file
     whichever you desire  \n\n")
21 nam=input('$ cat  ','s')
22 halt(' ')
23 clc
24 li(1)='#!/usr/bin/perl'
25 li(2)='# Script: '+nam+'.pl - Determines whether a
     year is leap year or not '
26 li(3)="#"
27 li(4)='die('+ascii(34)+'You have not entered the
     year\n'+ascii(34)+')  if (@ARGV == 0 ) ;'
28 li(5)='$year = $ARGV[0] ;
                                                    #
     The first argument'
29 li(6)='$last2digits = substr($year,-2,2) ;
                              #Extract from the right '
30 li(7)='if ($last2digits eq '+ascii(34)+'00'+ascii
     (34)+') {'
31 li(8)='          $yesorno = ($year % 400 == 0 ? '+
     ascii(34)+'certainly'+ascii(34)+' : '+ascii(34)+'
     not'+ascii(34)+' ) ;'
32 li(9)='}'
33 li(10)='else {'
34 li(11)='          $yesorno = ($year % 4 == 0 ? '+
     ascii(34)+'certainly'+ascii(34)+' : '+ascii(34)+'
     not'+ascii(34)+' ) ;'
35 li(12)=li(9)
36 li(13)='print('+ascii(34)+'$year is  '+ascii(34)+' .
     $yesorno . '+ascii(34)+' a leap year \n'+ascii
     (34)+') ;'
37
38 li(14)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
     '+ascii(39)+'to go back to console\n\n'+ascii(34)
     +')'
```

```
39 halt(' ')
40
41 v=mopen(nam+'.pl','wt')
42 for i=1:14
43     mfprintf(v,"%s\n",li(i))
44     if i~=14 then
45     printf("%s\n",li(i))
46     end
47 end
48 mclose(v)
49 if getos()=='Linux' then
50     printf("\n\nPlease open a new terminal window
          and then go to the directory %s and execute
          the following instruction\n\nperl %s.pl [
          Command line parameters if any]\n\nThank You
          \n\n",curr,nam)
51     halt(' ')
52     exit
53 end
54
55
56 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
57 printf(" \n       %c perl %s.pl [THE YEAR AS
     COMMANDLINE ARGUMENT]     %c[ENTER]\n\n",ascii(34)
     ,nam,ascii(34))
58
59 printf("\n$ perl %s.pl  [THE YEAR AS COMMANDLINE
     ARGUMENT]              #to execute the perlscript"
     ,nam)
60 halt(' ')
61 dos('start')
62 printf("\n\n\n")
63 halt('  ------------------>Executing PerlScript in
     Command Line Prompt<---------------- ')
64 printf("\n\n\n$ exit         #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
```

```
65 halt(".....# (hit [ENTER] for result)")
66 //clc()
67
68 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
69 sleep(1000)
70
71 mdelete(nam+'.pl')
```

Scilab code Exa 19.6 Program 6

```
1 clear
2 flag=1
3 mode(-1)
4 pwd
5 curr=ans
6 clc
7
8 printf("Example 6     :                Show the method
      of looping using %cforeach%c in perl \n",ascii
     (39),ascii(39))
9 disp("
     ************************************************************
     ")
10 disp("Answer     :    ")
11
12 disp("INSTRUCTIONS    : ")
13 printf("\n1. Here all instructions are preloaded in
     the form of a demo\n\nInitially the whole perl
     script is displaying and then \n the result of
     the same can be seen in the command line
```

```
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
14 halt('.............Press [ENTER] to continue.....')
15 halt("")
16 clc
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
18
19
20 printf("\n# Enter the name of the perlscript file
      whichever you desire  \n\n")
21 nam=input('$ cat  ','s')
22 halt(' ')
23 clc
24 li(1)='#!/usr/bin/perl'
25 li(2)='# Script: '+nam+'.pl  - Finds the square root
       of each command line argument  '
26 li(3)="#"
27 li(4)='print('+ascii(34)+'The program you are
      running is $0\n'+ascii(34)+')   ;'
28 li(5)='foreach  $number  (@ARGV)  {
                            # Each element of $ARGV
      goes to $number '
29 li(6)='        print('+ascii(34)+'The square root of
       $number is '+ascii(34)+' . sqrt($number) . '+
      ascii(34)+'\n'+ascii(34)+') ;'
30 li(7)='}'
31 li(8)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit'
      +ascii(39)+'to go back to console\n\n'+ascii(34)+
      ')'
32 halt(' ')
33
34 v=mopen(nam+'.pl','wt')
35 for i=1:8
```

```
36      mfprintf(v,"%s\n",li(i))
37      if i~=8 then
38      printf("%s\n",li(i))
39      end
40  end
41  mclose(v)
42  if getos()=='Linux' then
43      printf("\n\nPlease open a new terminal window
           and then go to the directory %s and execute
           the following instruction\n\nperl %s.pl [
           Command line parameters if any]\n\nThank You
           \n\n",curr,nam)
44      halt(' ')
45      exit
46  end
47
48  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
49  printf(" \n       %c perl %s.pl [THE NUMBERS AS
       COMMANDLINE ARGUMENTS]     %c[ENTER]\n\n",ascii
       (34),nam,ascii(34))
50
51  printf("\n$ perl %s.pl  [THE NUMBERS AS COMMANDLINE
       ARGUMENTS]              #to execute the perlscript
       ",nam)
52  halt(' ')
53  dos('start')
54  printf("\n\n\n")
55  halt('  ------------------>Executing PerlScript in
       Command Line Prompt<---------------- ')
56  printf("\n\n\n$ exit         #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
57  halt("........# (hit [ENTER] for result)")
58  //clc()
59
60  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
```

```
61 sleep(1000)
62
63 mdelete(nam+'.pl')
```

Scilab code Exa 19.7 Program 7

```
1  clear
2  mode(-1)
3  pwd
4  curr=ans
5  clc
6
7  printf("Example 7     :                Show the method
       of  splititng a string using %csplit%c in perl \
     n",ascii(39),ascii(39))
8  disp("
     ************************************************************
     ")
9  disp("Answer     :    ")
10
11 disp("INSTRUCTIONS    : ")
12 printf("\n1. Here all instructions are preloaded in
     the form of a demo\n\nInitially the whole perl
     script is displaying and then \n the result of
     the same can be seen in the command line
     interpreter.\n\n2. PLEASE MAKE SURE THAT THE
     PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
     THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
     AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
     ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
     \n")
```

```
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
17
18
19 printf("\n# Enter the name of the perlscript file
     whichever you desire  \n\n")
20 nam=input('$ cat  ','s')
21 halt(' ')
22 clc
23 li(1)='#!/usr/bin/perl'
24 li(2)='# Script: '+nam+'.pl  - Finds the square root
     of each command line argument  '
25 li(3)="#"
26 li(4)='print('+ascii(34)+'Enter three numbers: '+
     ascii(34)+' )  ; '
27 li(5)='chop($numstring=<STDIN>) ;'
28 li(6)='die('+ascii(34)+'Nothing entered\n'+ascii(34)
     +')  if ($numstring eq '+ascii(34)+ascii(34)+' )
      ;'
29 li(7)='($f_number,$s_number,$l_number) = split(/ /,
     $numstring)  ;'
30 li(8)='print('+ascii(34)+'The last, second and first
     numbers are '+ascii(34)+')  ;'
31 li(9)='print('+ascii(34)+'$l_number,$s_number and
     $f_number.\n'+ascii(34)+')  ;'
32
33 li(10)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
     '+ascii(39)+'to go back to console\n\n'+ascii(34)
     +')'
34 halt(' ')
35
36 v=mopen(nam+'.pl','wt')
37 for i=1:10
38     mfprintf(v,"%s\n",li(i))
39     if i~=10 then
```

```
40      printf("%s\n",li(i))
41      end
42  end
43  mclose(v)
44  if getos()=='Linux' then
45      printf("\n\nPlease open a new terminal window
           and then go to the directory %s and execute
           the following instruction\n\nperl %s.pl [
           Command line parameters if any]\n\nThank You
           \n\n",curr,nam)
46      halt(' ')
47      exit
48  end
49
50
51  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
52  printf(" \n       %c perl %s.pl     %c[ENTER]\n\n",
       ascii(34),nam,ascii(34))
53
54  printf("\n$ perl %s.pl                #to execute the
        perlscript",nam)
55  halt(' ')
56  dos('start')
57  printf("\n\n\n")
58  halt('  ------------------>Executing PerlScript in
       Command Line Prompt<---------------- ')
59  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
60  halt("........# (hit [ENTER] for result)")
61  //clc()
62
63  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
        initial environment')
64  sleep(1000)
65
66  mdelete(nam+'.pl')
```

Scilab code Exa 19.8 Program 8

```
1  clear
2  mode(-1)
3  pwd
4  curr=ans
5  clc
6
7  printf("Example 8     :                Show the method
       of  splitting a string to an array using
      %csplit%c in perl \n",ascii(39),ascii(39))
8  disp("
      ************************************************************
      ")
9  disp("Answer     :    ")
10
11
12 disp("INSTRUCTIONS    : ")
13 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n\n6. Afile named %cemp.lst%c with the necessary
      details gets created automatically for the
      session",ascii(34),ascii(34))
```

```
14 halt('.............Press [ENTER] to continue.....')
15 halt("")
16 clc
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
18
19 i=0
20 i=i+1;f(i)='2233|a.k.shukla                |g.m.
            |sales            |12/12/52|6000'
21 i=i+1;f(i)='9876|jai sharma                |director   |
      production |12/03/50|7000'
22 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m       |
      marketing  |19/04/43|6000'
23 i=i+1;f(i)='2356|barun sengupta       |director   |
      personnel  |11/05/47|7800'
24 i=i+1;f(i)='5423|n.k. gupta              |chairman |
      admin       |30/08/56|5400'
25 i=i+1;f(i)='1006|chanchal singhvi    |director   |
      sales          |03/09/38|6700'
26 i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
      accounts   |05/06/62|6300'
27 i=i+1;f(i)='1265|s.n. dasgupta          |manager  |
      sales          |12/09/63|5600'
28 i=i+1;f(i)='4290|jayant Choudhary  |executive |
      production|07/09/50|6000'
29 i=i+1;f(i)='2476|anil aggarwal         |manager    |
      sales          |01/05/59|5000'
30 i=i+1;f(i)='6521|lalit chowdury        |director     |
      marketing |26/09/45|8200'
31 i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
      accounts   |12/12/55|6000'
32 i=i+1;f(i)='3564|sudhir Agarwal      |executive  |
      personnel  |06/07/47|7500'
33 i=i+1;f(i)='2345|j.b. saxena              |g.m.
               |marketing |12/03/45|8000'
34 i=i+1;f(i)='0110|v.k. agrawal          |g.m.
               |marketing  |31/02/40|9000'
35 n=i
```

```
36 printf("\n\n$ cat emp.lst      # to open the file
     emp.lst")
37 halt(' ')
38 u=mopen('emp.lst','wt')
39 for i=1:n
40     mfprintf(u,"%s\n",f(i))
41     printf("%s\n",f(i))
42 end
43 mclose(u)
44
45 printf("\n# Enter the name of the perlscript file
     whichever you desire  \n\n")
46 nam=input('$ cat  ','s')
47 halt(' ')
48 clc
49 li(1)='#!/usr/bin/perl'
50 li(2)='# Script: '+nam+'.pl  - Uses split twice;
     prints with first and last name reversed  '
51 li(3)="#"
52 li(4)='while (<>)  {'
53 li(5)='            chop; '
54 li(6)='             @field = split(/\|/) ;          #
     $_ is used by default '
55 li(7)='              if (1..4)    {
                          # Lines 1 to 4 '
56 li(8)='                $dept = $field[3]  ;
     $name = $field[1]  ; $salary = $field[5]  ;'
57 li(9)='                ($f_name,$l_name) = split
     (/ +/,$name) ; '
58 li(10)='                 $name = $l_name . '+ascii
     (34)+',  '+ascii(34)+' .  $f_name  ;     #Reusing
      $name'
59 li(11)='                 $totsal += $salary ;'
60 li(12)='                 printf('+ascii(34)+'%3d %
     -20s %-11s %4d\n'+ascii(34)+', $. ,$name , $dept
     ,$salary )   ;'
61 li(13)='     }'
62 li(14)=' }'
```

```
63 li(15)='printf('+ascii(34)+'%35s %5d\n'+ascii(34)+',
     '+ascii(34)+'Total Salary: '+ascii(34)+',
     $totsal)    ;'
64
65
66 li(16)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
     '+ascii(39)+'to go back to console\n\n'+ascii(34)
     +')'
67 halt(' ')
68
69 v=mopen(nam+'.pl','wt')
70 for i=1:16
71     mfprintf(v,"%s\n",li(i))
72     if i~=16 then
73     printf("%s\n",li(i))
74     end
75 end
76 mclose(v)
77 if getos()=='Linux' then
78     printf("\n\nPlease open a new terminal window
         and then go to the directory %s and execute
         the following instruction\n\nperl %s.pl [
         Command line parameters if any]\n\nThank You
         \n\n",curr,nam)
79     halt(' ')
80     exit
81 end
82
83
84 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
85 printf(" \n       %c %s.pl  emp.lst   %c[ENTER]\n\n"
     ,ascii(34),nam,ascii(34))
86
87 printf("\n$ %s.pl   emp.lst            #to execute
     the perlscript",nam)
88 halt(' ')
89 dos('start')
```

```
90 printf("\n\n\n")
91 halt('   ------------------>Executing PerlScript in
      Command Line Prompt<---------------- ')
92 printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
93 halt(".......#  (hit [ENTER] for result)")
94 //clc()
95
96 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
97 sleep(1000)
98
99 mdelete(nam+'.pl')
100 mdelete('emp.lst')
```

Scilab code Exa 19.9 Program 9

```
1 clear
2 mode(-1)
3 pwd
4 curr=ans
5 clc
6 //Program for example 1 chapter 1
7
8 printf("Example 9    :                Show the method
      of  joining an array using %cjoin%c in perl \n",
     ascii(39),ascii(39))
9 disp("
     ************************************************************
     ")
```

```
10 disp("Answer     :    ")
11
12 disp("INSTRUCTIONS    : ")
13 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n\n6.A file named %cemp.lst%c with the necessary
       details gets created automatically for the
      session",ascii(34),ascii(34))
14 halt('.............Press [ENTER] to continue.....')
15 halt("")
16 clc
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
18
19 i=0
20 i=i+1;f(i)='2233|a.k.shukla              |g.m.
            |sales          |12/12/52|6000'
21 i=i+1;f(i)='9876|jai sharma              |director   |
      production |12/03/50|7000'
22 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m      |
      marketing  |19/04/43|6000'
23 i=i+1;f(i)='2356|barun sengupta       |director   |
      personnel  |11/05/47|7800'
24 i=i+1;f(i)='5423|n.k. gupta              |chairman |
      admin       |30/08/56|5400'
25 i=i+1;f(i)='1006|chanchal singhvi    |director   |
      sales         |03/09/38|6700'
26 i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
      accounts   |05/06/62|6300'
27 i=i+1;f(i)='1265|s.n. dasgupta          |manager   |
      sales         |12/09/63|5600'
```

```
28 i=i+1;f(i)='4290|jayant Choudhary   |executive |
      production|07/09/50|6000'
29 i=i+1;f(i)='2476|anil aggarwal        |manager    |
      sales           |01/05/59|5000'
30 i=i+1;f(i)='6521|lalit chowdury       |director     |
      marketing |26/09/45|8200'
31 i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
      accounts    |12/12/55|6000'
32 i=i+1;f(i)='3564|sudhir Agarwal      |executive    |
      personnel   |06/07/47|7500'
33 i=i+1;f(i)='2345|j.b. saxena             |g.m.
                 |marketing |12/03/45|8000'
34 i=i+1;f(i)='0110|v.k. agrawal          |g.m.
                 |marketing   |31/02/40|9000'
35 n=i
36 printf("\n\n$ cat emp.lst       # to open the file
      emp.lst")
37 halt(' ')
38 u=mopen('emp.lst','wt')
39 for i=1:n
40     mfprintf(u,"%s\n",f(i))
41     printf("%s\n",f(i))
42 end
43 mclose(u)
44
45 printf("\n# Enter the name of the perlscript file
      whichever you desire  \n\n")
46 nam=input('$ cat  ','s')
47 halt(' ')
48 clc
49 li(1)='#!/usr/bin/perl  -n '
50 li(2)='# Script: '+nam+'.pl  - Uppercases the name
      and adds century  prefix to the date '
51 li(3)="#"
52 li(4)='@line = split(/\|/) ;

     # $_ is assumed '
53 li(5)='($day, $month, $year) =  split(/\// , $line
```

```
    [4]);                                                    #
      Splits date field'
li(6)='$year = '+ascii(34)+'19'+ascii(34)+' . $year
    ;                                                        #
    Adds century prefix'
li(7)='$line[4] = join('+ascii(34)+'\/'+ascii(34)+',
     $day , $month , $year) ;      #  Rebuilds date
    field'
li(8)='$line = join('+ascii(34)+'\|'+ascii(34)+',
    @line);                                                  #
    Rebuilds line'
li(9)='print  $line; '
li(10)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
    '+ascii(39)+'to go back to console\n\n'+ascii(34)
    +')'
halt(' ')

v=mopen(nam+'.pl','wt')
for i=1:9
    mfprintf(v,"%s\n",li(i))
    if i~=10 then
    printf("%s\n",li(i))
    end
end
mclose(v)
if getos()=='Linux' then
    printf("\n\nPlease open a new terminal window
       and then go to the directory %s and execute
       the following instruction\n\nperl %s.pl [
       Command line parameters if any]\n\nThank You
       \n\n",curr,nam)
    halt(' ')
    exit
end


printf("\n# type the following command in the
   command line interpreter as soon as it appears")
```

```
77 printf(" \n        %c %s.pl  emp.lst    %c[ENTER]\n\n"
   ,ascii(34),nam,ascii(34))
78
79 printf("\n$ %s.pl    emp.lst              #to execute
   the perlscript",nam)
80 halt(' ')
81 dos('start')
82 printf("\n\n\n")
83 halt('   ------------------>Executing PerlScript in
   Command Line Prompt<---------------- ')
84 printf("\n\n\n$ exit          #To exit the current
   simulation terminal and return to Scilab console\
   n\n")
85 halt("........# (hit [ENTER] for result)")
86 //clc()
87
88 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
89 sleep(1000)
90
91 mdelete(nam+'.pl')
92 mdelete('emp.lst')
```

Scilab code Exa 19.10 Program 10

```
1 clear
2 mode(-1)
3 pwd
4 curr=ans
5 clc
6
```

```
7  printf("Example 10     :                    Show the
       method of converting a decimal to binary in perl
       \n")
8  disp("
       ************************************************************
       ")
9  disp("Answer     :    ")
10
11 disp("INSTRUCTIONS    : ")
12 printf("\n1. Here all instructions are preloaded in
       the form of a demo\n\nInitially the whole perl
       script is displaying and then \n the result of
       the same can be seen in the command line
       interpreter.\n\n2. PLEASE MAKE SURE THAT THE
       PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
       THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
       AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
       ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
       \n")
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
       PRELOADED COMMANDS)\n\n\n")
17
18
19 printf("\n# Enter the name of the perlscript file
       whichever you desire  \n\n")
20 nam=input('$ cat  ','s')
21 halt(' ')
22 clc
23 li(1)='#!/usr/bin/perl'
24 li(2)='# Script: '+nam+'.pl - Converts decimal
       numbers to binary '
25 li(3)="#"
26 li(4)='die('+ascii(34)+'You have not entered any
       number\n'+ascii(34)+')  if (@ARGV == 0 ) ;'
27 li(5)='foreach $number (@ARGV) { '
```

```
li(6)='          $original_number = $number ;'
li(7)='          until ($number == 0)  {'
li(8)='                    $bit = $number % 2 ;
                              #  Find the remainder
    bit'
li(9)='                    unshift (@bit_arr , $bit )
     ;                   #  Insert bit at beginning'
li(10)='                   $number = int($number / 2) ;
     '
li(11)='         }'
li(12)='         $binary_number = join ( '+ascii(34)+
    ascii(34)+', @bit_arr)  ;   # Join or nothing!'
li(13)='         print('+ascii(34)+'The binary number
    of $original_number is $binary_number\n'+ascii
    (34)+')   ;'
li(14)='         $#bit_arr = -1 ; }
                                                #deletes
     all array elements'
li(15)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
    '+ascii(39)+'to go back to console\n\n'+ascii(34)
    +')'
halt(' ')

v=mopen(nam+'.pl','wt')
for i=1:14
    mfprintf(v,"%s\n",li(i))
    if i~=14 then
    printf("%s\n",li(i))
    end
end
mclose(v)
if getos()=='Linux' then
    printf("\n\nPlease open a new terminal window
       and then go to the directory %s and execute
       the following instruction\n\nperl %s.pl [
       Command line parameters if any]\n\nThank You
       \n\n",curr,nam)
    halt(' ')
```

```
51      exit
52  end
53
54  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
55  printf(" \n        %c %s.pl [THE NUMBERS AS
       COMMANDLINE ARGUMENTS]     %c[ENTER]\n\n",ascii
       (34),nam,ascii(34))
56
57  printf("\n$  %s.pl  [THE NUMBERS AS COMMANDLINE
       ARGUMENTS]              #to execute the perlscript
       ",nam)
58  halt(' ')
59  dos('start')
60  printf("\n\n\n")
61  halt('   ------------------->Executing PerlScript in
       Command Line Prompt<---------------- ')
62  printf("\n\n\n$ exit         #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
63  halt(".......# (hit [ENTER] for result)")
64  //clc()
65
66  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
67  sleep(1000)
68
69  mdelete(nam+'.pl')
```

Scilab code Exa 19.11 Program 11

```
clear
mode(-1)
pwd
curr=ans
clc

printf("Example 11     :                    Show the
   method of  searching in an array using %cgrep%c
   in perl \n",ascii(39),ascii(39))
disp("
   ************************************************************
   ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
   AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
   ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
   \n\n6.A file named %cdept.lst%c with the
   necessary details gets created automatically for
   the session",ascii(34),ascii(34))
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
   PRELOADED COMMANDS)\n\n\n")

i=0
i=i+1;f(i)='01|accounts|6213'
i=i+1;f(i)='02|admin|5423'
i=i+1;f(i)='03|marketing|6521'
i=i+1;f(i)='04|personnel|2365'
```

```
23 i=i+1;f(i)='05|production|9876'
24 i=i+1;f(i)='06|sales|1006'
25 n=i
26 printf("\n\n$ cat dept.lst      # to open the file
     dept.lst")
27 halt(' ')
28 u=mopen('dept.lst','wt')
29 for i=1:n
30     mfprintf(u,"%s\n",f(i))
31     printf("%s\n",f(i))
32 end
33 mclose(u)
34
35 printf("\n# Enter the name of the perlscript file
     whichever you desire  \n\n")
36 nam=input('$ cat  ','s')
37 clc
38 li(1)='#!/usr/bin/perl'
39 li(2)='# Script: '+nam+'.pl  - Searches array for a
     string or regular expression  '
40 li(3)="#"
41 li(4)='@dept_arr = <> ;
                                              #  Read
     file into array'
42 li(5)='for ($i=0 ; $i<3 ; $i++) {
                          #  Can use only three times
     ,
43 li(6)='       print('+ascii(34)+'Enter a code to
     look up: '+ascii(34)+') ;  '
44 li(7)='       chop($code = <STDIN>)  ;'
45 li(8)='       @found_arr = grep (/^$code/, @dept_arr
     )  ;          #  Search at beginning'
46 li(9)='         if  ($#found_arr == -1 || $code eq '
     +ascii(34)+ascii(34)+') {       #  -1 means null
     array'
47 li(10)='                    print('+ascii(34)+'Code
     does not exist\n'+ascii(34)+') ; '
48 li(11)='                    next ;
```

```
                                                    #  Go to the
    beginning of loop'
49  li(12)='     }'
50  li(13)='     @tt=split(/\|/ ,$found_arr[0])  ;
                        #  Split first element only '
51  li(14)='       print('+ascii(34)+'Code = $code
    Description = $tt[1]\n'+ascii(34)+') ;  '
52  li(15)=' }'
53
54
55  li(16)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
    '+ascii(39)+'to go back to console\n\n'+ascii(34)
    +')'
56  halt(' ')
57
58  printf("\n$ cat %s.pl                #open the
    perlscript file %s.pl\n\n",nam,nam)
59  v=mopen(nam+'.pl','wt')
60  for i=1:16
61      mfprintf(v,"%s\n",li(i))
62      if i~=16 then
63      printf("%s\n",li(i))
64      end
65  end
66  mclose(v)
67  if getos()=='Linux' then
68      printf("\n\nPlease open a new terminal window
           and then go to the directory %s and execute
           the following instruction\n\nperl %s.pl [
           Command line parameters if any]\n\nThank You
           \n\n",curr,nam)
69      halt(' ')
70      exit
71  end
72
73  printf("\n# type the following command in the
    command line interpreter as soon as it appears")
74  printf(" \n       %c %s.pl  dept.lst   %c[ENTER]\n\n
```

```
    ",ascii(34),nam,ascii(34))
75
76 printf("\n$ %s.pl   dept.lst              #to execute
    the perlscript",nam)
77 halt(' ')
78 dos('start')
79 printf("\n\n\n")
80 halt('   ------------------>Executing PerlScript in
    Command Line Prompt<---------------- ')
81 printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
82 halt(".......# (hit [ENTER] for result)")
83 //clc()
84
85 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
86 sleep(1000)
87
88 mdelete(nam+'.pl')
89 mdelete('dept.lst')
```

Scilab code Exa 19.12 Program 12

```
1 clear
2 mode(-1)
3 pwd
4 curr=ans
5 clc
6
7 printf("Example 12     :                Show the
```

```
    method of  using %cASSOCIATIVE ARRAYS%c in perl \
    n",ascii(39),ascii(39))
8 disp("
    **********************************************************
    ")
9 disp("Answer     :    ")
10
11 disp("INSTRUCTIONS    : ")
12 printf("\n1. Here all instructions are preloaded in
    the form of a demo\n\nInitially the whole perl
    script is displaying and then \n the result of
    the same can be seen in the command line
    interpreter.\n\n2. PLEASE MAKE SURE THAT THE
    PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
    THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
    AFTER EACH COMMAND to see its RESULT\n\n4. PRESS
    ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
    \n")
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
17
18
19 printf("\n# Enter the name of the perlscript file
    whichever you desire  \n\n")
20 nam=input('$ cat  ','s')
21 halt(' ')
22 clc
23 printf("\n$ cat %s        #to open perlscript file
    %s  \n ",nam,nam)
24 li(1)='#!/usr/bin/perl'
25 li(2)='# Script: '+nam+'.pl  - Uses an associative
    array  '
26 li(3)="#"
27 li(4)='%region = ('+ascii(34)+'N'+ascii(34)+','+
    ascii(34)+'North'+ascii(34)+','+ascii(34)+'S'+
```

```
      ascii(34)+',"+ascii(34)+'South'+ascii(34)+',"+
      ascii(34)+'E'+ascii(34)+',"+ascii(34)+'East'+
      ascii(34)+',"+ascii(34)+'W'+ascii(34)+',"+ascii
      (34)+'West'+ascii(34)+') ;'
28 li(5)='die('+ascii(34)+'Nothing entered in
      commandline\n'+ascii(34)+')  if (@ARGV == 0 )  ;'
29 li(6)='foreach $letter (@ARGV) {'
30 li(7)='    print('+ascii(34)+'The letter $letter
      stands for $region{$letter}'+ascii(34)+'.'+ascii
      (34)+'\n'+ascii(34)+')  ; '
31 li(8)='}'
32 li(9)='@key_list = keys(%region)  ;

     #  List of subscripts'
33 li(10)='print('+ascii(34)+'The subscripts are
      @key_list\n'+ascii(34)+') ;  '
34 li(11)='@value_list = values %region    ;
                                                #
      List of values'
35 li(12)='print('+ascii(34)+'The values are
      @value_list\n'+ascii(34)+') ;  '
36
37 li(13)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
      '+ascii(39)+'to go back to console\n\n'+ascii(34)
      +')'
38 halt(' ')
39
40 v=mopen(nam+'.pl','wt')
41 for i=1:13
42     mfprintf(v,"%s\n",li(i))
43     if i~=13 then
44     printf("%s\n",li(i))
45     end
46 end
47 mclose(v)
48 if getos()=='Linux' then
49     printf("\n\nPlease open a new terminal window
         and then go to the directory %s and execute
```

```
        the following instruction\n\nperl %s.pl [
        Command line parameters if any]\n\nThank You
        \n\n",curr,nam)
50      halt(' ')
51      exit
52  end
53
54
55  printf("\n# type the following command in the
    command line interpreter as soon as it appears")
56  printf(" \n       %c perl %s.pl     %c[ENTER]\n\n",
    ascii(34),nam,ascii(34))
57
58  printf("\n$ perl %s.pl                #to execute the
     perlscript",nam)
59  halt(' ')
60  dos('start')
61  printf("\n\n\n")
62  halt('   ------------------>Executing PerlScript in
    Command Line Prompt<---------------- ')
63  printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
64  halt(".......# (hit [ENTER] for result)")
65  //clc()
66
67  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
68  sleep(1000)
69
70  mdelete(nam+'.pl')
```

Scilab code Exa 19.13 Program 13

```
1  clear
2  mode(-1)
3  pwd
4  curr=ans
5  clc
6
7  printf("Example 13     :                Show the
      method of finding the frequency of occurence in
      perl \n")
8  disp("
      ************************************************************
      ")
9  disp("Answer     :    ")
10
11 disp("INSTRUCTIONS    : ")
12 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n\n6. Afile named %cemp.lst%c with the necessary
      details gets created automatically for the
      session",ascii(34),ascii(34))
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
17
18 i=0
19 i=i+1;f(i)='2233|a.k.shukla                |g.m.
            |sales|12/12/52|6000'
```

```
20 i=i+1;f(i)='9876|jai sharma              |director   |
     production|12/03/50|7000'
21 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m       |
     marketing|19/04/43|6000'
22 i=i+1;f(i)='2356|barun sengupta       |director   |
     personnel|11/05/47|7800'
23 i=i+1;f(i)='5423|n.k. gupta              |chairman |
     admin|30/08/56|5400'
24 i=i+1;f(i)='1006|chanchal singhvi    |director   |
     sales|03/09/38|6700'
25 i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
     accounts|05/06/62|6300'
26 i=i+1;f(i)='1265|s.n. dasgupta          |manager   |
     sales|12/09/63|5600'
27 i=i+1;f(i)='4290|jayant Choudhary   |executive |
     production|07/09/50|6000'
28 i=i+1;f(i)='2476|anil aggarwal         |manager    |
     sales|01/05/59|5000'
29 i=i+1;f(i)='6521|lalit chowdury        |director     |
     marketing|26/09/45|8200'
30 i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
     accounts|12/12/55|6000'
31 i=i+1;f(i)='3564|sudhir Agarwal      |executive   |
     personnel|06/07/47|7500'
32 i=i+1;f(i)='2345|j.b. saxena             |g.m.
                |marketing|12/03/45|8000'
33 i=i+1;f(i)='0110|v.k. agrawal           |g.m.
                |marketing|31/02/40|9000'
34 n=i
35 printf("\n\n$ cat emp.lst       # to open the file
     emp.lst")
36 halt(' ')
37 u=mopen('emp.lst','wt')
38 for i=1:n
39     mfprintf(u,"%s\n",f(i))
40     printf("%s\n",f(i))
41 end
42 mclose(u)
```

```
43
44 printf("\n# Enter the name of the perlscript file
      whichever you desire  \n\n")
45 nam=input('$ cat  ','s')
46 clc
47 li(1)='#!/usr/bin/perl'
48 li(2)='# Script: '+nam+'.pl  - Counts frequency of
      occurence of an item  '
49 li(3)="#"
50 li(4)='while (<>)  {'
51 li(5)='             @t=split(/\|/) ;
                              #  | has to be escaped'
52 li(6)='             $dept= $t[3]  ;                #
       Department is fourth field'
53 li(7)='             $deptlist{$dept}++ ;  #  same as
      ++'
54 li(8)='     }'
55 li(9)='foreach $det (sort (keys %deptlist))   {'
56 li(10)='      print('+ascii(34)+'$det: $deptlist{
      $det}\n'+ascii(34)+') ; '
57 li(11)=' }'
58
59
60
61 li(12)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
      '+ascii(39)+'to go back to console\n\n'+ascii(34)
      +')'
62 halt(' ')
63
64 printf("\n $ cat %s        # to open the perlscript
      file %s ",nam,nam)
65 v=mopen(nam+'.pl','wt')
66 for i=1:12
67     mfprintf(v,"%s\n",li(i))
68     if i~=12 then
69     printf("%s\n",li(i))
70     end
71 end
```

```
72 mclose(v)
73 if getos()=='Linux' then
74     printf("\n\nPlease open a new terminal window
          and then go to the directory %s and execute
          the following instruction\n\nperl %s.pl [
          Command line parameters if any]\n\nThank You
          \n\n",curr,nam)
75     halt(' ')
76     exit
77 end
78
79
80 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
81 printf(" \n       %c %s.pl  emp.lst   %c[ENTER]\n\n"
     ,ascii(34),nam,ascii(34))
82
83 printf("\n$ %s.pl   emp.lst            #to execute
     the perlscript",nam)
84 halt(' ')
85 dos('start')
86 printf("\n\n\n")
87 halt('  ------------------>Executing PerlScript in
     Command Line Prompt<---------------- ')
88 printf("\n\n\n$ exit        #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
89 halt("........# (hit [ENTER] for result)")
90 //clc()
91
92 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
93 sleep(1000)
94
95 mdelete(nam+'.pl')
96 mdelete('emp.lst')
```

Scilab code Exa 19.14 Program 14

```
clear
mode(-1)
pwd
curr=ans
clc

printf("Example 14     :                Show the
   method of  using regular expressions in perl \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
   AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
   ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
   \n\n6. Afile named %cemp.lst%c with the necessary
   details gets created automatically for the
   session",ascii(34),ascii(34))
halt('.............Press [ENTER] to continue.....')
halt("")
```

```
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
17
18 i=0
19 i=i+1;f(i)='2233|a.k.shukla                |g.m.
            |sales            |12/12/52|6000'
20 i=i+1;f(i)='9876|jai sharma                |director   |
      production |12/03/50|7000'
21 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m       |
      marketing   |19/04/43|6000'
22 i=i+1;f(i)='2356|barun sengupta       |director   |
      personnel   |11/05/47|7800'
23 i=i+1;f(i)='5423|n.k. gupta                |chairman |
      admin        |30/08/56|5400'
24 i=i+1;f(i)='1006|chanchal singhvi    |director   |
      sales           |03/09/38|6700'
25 i=i+1;f(i)='6213|karuna ganguly       |g.m.         |
      accounts    |05/06/62|6300'
26 i=i+1;f(i)='1265|s.n. dasgupta          |manager   |
      sales           |12/09/63|5600'
27 i=i+1;f(i)='4290|jayant Choudhary   |executive |
      production|07/09/50|6000'
28 i=i+1;f(i)='2476|anil aggarwal         |manager    |
      sales           |01/05/59|5000'
29 i=i+1;f(i)='6521|lalit chowdury        |director     |
      marketing |26/09/45|8200'
30 i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
      accounts    |12/12/55|6000'
31 i=i+1;f(i)='3564|sudhir Agarwal      |executive   |
      personnel   |06/07/47|7500'
32 i=i+1;f(i)='2345|j.b. saxena              |g.m.
               |marketing |12/03/45|8000'
33 i=i+1;f(i)='0110|v.k. agrawal           |g.m.
               |marketing   |31/02/40|9000'
34 n=i
35 printf("\n\n$ cat emp.lst       # to open the file
      emp.lst")
```

```
36 halt(' ')
37 u=mopen('emp.lst','wt')
38 for i=1:n
39     mfprintf(u,"%s\n",f(i))
40     printf("%s\n",f(i))
41 end
42 mclose(u)
43
44 printf("\n# Enter the name of the perlscript file
      whichever you desire  \n\n")
45 nam=input('$ cat  ','s')
46 halt(' ')
47 clc
48 li(1)='#!/usr/bin/perl'
49 li(2)='# Script: '+nam+'.pl  -  Uses s and tr
      functions for substitution  '
50 li(3)="#"
51 li(4)='print('+ascii(34)+'Last two digits of date of
       birth:  '+ascii(34)+');'
52 li(5)='$yearin = <STDIN> ;'
53 li(6)='chop($yearin);
                                              #  Remove \n
      else comparison will fail later'
54 li(7)='   '
55 li(8)='$found = 0;'
56 li(9)='while (<>)  {'
57 li(10)='             @line = split(/\|/) ;          '
58 li(11)='               $name = $line[1] ; $emp_id =
      $line[0]   ; '
59 li(12)='              @tt=split(/\//,$line[4]);
              #Splits date field'
60 li(13)='               $year = $tt[2] ;
                          #2-digit year extracted ..'
61 li(14)=li(7)
62 li(15)='               if($year eq $yearin)   {
          # .. and compared with the user input'
63 li(16)='                           $found = 1;'
64 li(17)='                           $name =~ tr/a-z/A-Z
```

```
      / ;   # Name field changed to caps'
65  li(18)='                                $emp_id =~ s/^/9/
        ;       # Adds 9 as a prefix to employee id'
66  li(19)='                                $line[0] = $emp_id
       ;      #Reassign'
67  li(20)='                                $line[1] =  $name
      ;          #with changes'
68  li(21)='                                 $x=join('+ascii
      (34)+':'+ascii(34)+', @line) ;'
69  li(22)='                                $x=~s/\s+:/:/g  ;
                          #Removes whitespace before
      delimiter'
70  li(23)='                                $x=~s#/#-#g     ;
                             #New delimiter in date'
71  li(24)='                                print $x; } }'
72  li(25)='print('+ascii(34)+'Year 19'+ascii(34)+'.
      $yearin . '+ascii(34)+'not found\n'+ascii(34)+')
        if $found eq 0  ;'
73
74
75  li(26)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
      '+ascii(39)+'to go back to console\n\n'+ascii(34)
      +')'
76  halt(' ')
77
78  v=mopen(nam+'.pl','wt')
79  for i=1:26
80      mfprintf(v,"%s\n",li(i))
81      if i~=26 then
82      printf("%s\n",li(i))
83      end
84  end
85  mclose(v)
86  if getos()=='Linux' then
87      printf("\n\nPlease open a new terminal window
          and then go to the directory %s and execute
          the following instruction\n\nperl %s.pl [
          Command line parameters if any]\n\nThank You
```

```
        \n\n",curr,nam)
88      halt(' ')
89      exit
90  end
91
92  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
93  printf(" \n        %c %s.pl  emp.lst   %c[ENTER]\n\n"
       ,ascii(34),nam,ascii(34))
94
95  printf("\n$ %s.pl   emp.lst              #to execute
       the perlscript",nam)
96  halt(' ')
97  dos('start')
98  printf("\n\n\n")
99  halt('   ------------------>Executing PerlScript in
       Command Line Prompt<---------------- ')
100 printf("\n\n\n$ exit         #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
101 halt("........# (hit [ENTER] for result)")
102 //clc()
103
104 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
105 sleep(1000)
106
107 mdelete(nam+'.pl')
108 mdelete('emp.lst')
```

Scilab code Exa 19.15 Program 15

```
clear
pwd
curr=ans
mode(-1)
clc

printf("Example 15     :                Show the
   method of  using TRE and IRE in perl \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
   the form of a demo\n\nInitially the whole perl
   script is displaying and then \n the result of
   the same can be seen in the command line
   interpreter.\n\n2. PLEASE MAKE SURE THAT THE
   PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
   THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
   AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
   ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
   \n\n6. Afile named %cemp.lst%c with the necessary
   details gets created automatically for the
   session",ascii(34),ascii(34))
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
   PRELOADED COMMANDS)\n\n\n")

i=0
i=i+1;f(i)='2233|a.k.shukla              |g.m.
          |sales          |12/12/52|6000'
i=i+1;f(i)='9876|jai sharma              |director   |
   production |12/03/50|7000'
i=i+1;f(i)='5678|sumit chakrobarty|d.g.m       |
```

```
    marketing   |19/04/43|6000'
22  i=i+1;f(i)='2356|barun sengupta       |director   |
    personnel   |11/05/47|7800'
23  i=i+1;f(i)='5423|n.k. gupta              |chairman |
    admin       |30/08/56|5400'
24  i=i+1;f(i)='1006|chanchal singhvi    |director   |
    sales           |03/09/38|6700'
25  i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
    accounts    |05/06/62|6300'
26  i=i+1;f(i)='1265|s.n. dasgupta          |manager   |
    sales           |12/09/63|5600'
27  i=i+1;f(i)='4290|jayant Choudhary   |executive |
    production|07/09/50|6000'
28  i=i+1;f(i)='2476|anil aggarwal         |manager    |
    sales           |01/05/59|5000'
29  i=i+1;f(i)='6521|lalit chowdury        |director     |
    marketing |26/09/45|8200'
30  i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
    accounts    |12/12/55|6000'
31  i=i+1;f(i)='3564|sudhir Agarwal      |executive   |
    personnel   |06/07/47|7500'
32  i=i+1;f(i)='2345|j.b. saxena             |g.m.
                |marketing |12/03/45|8000'
33  i=i+1;f(i)='0110|v.k. agrawal          |g.m.
                |marketing   |31/02/40|9000'
34  n=i
35  printf("\n\n$ cat emp.lst      # to open the file
    emp.lst")
36  halt(' ')
37  u=mopen('emp.lst','wt')
38  for i=1:n
39      mfprintf(u,"%s\n",f(i))
40      printf("%s\n",f(i))
41  end
42  mclose(u)
43
44  printf("\n# Enter the name of the perlscript file
    whichever you desire  \n\n")
```

```
45 nam=input('$ cat  ','s')
46 halt(' ')
47 clc
48 li(1)='#!/usr/bin/perl -n'
49 li(2)='# Script: '+nam+'.pl  - Reports a date in
     format dd-mm-yyyy using a TRE  '
50 li(3)="#"
51 li(4)='@month[1..12] = ('+ascii(34)+'Jan'+ascii(34)+
     ','+ascii(34)+'Feb'+ascii(34)+','+ascii(34)+'Mar'
     +ascii(34)+','+ascii(34)+'Apr'+ascii(34)+','+
     ascii(34)+'May'+ascii(34)+','+ascii(34)+'Jun'+
     ascii(34)+','+ascii(34)+'Jul'+ascii(34)+','+ascii
     (34)+'Aug'+ascii(34)+','+ascii(34)+'Sep'+ascii
     (34)+','+ascii(34)+'Oct'+ascii(34)+','+ascii(34)+
     'Nov'+ascii(34)+','+ascii(34)+'Dec'+ascii(34)+')
     ;'
52 li(5)='@x = split(/\|/) ;'
53 li(6)='$x[4] =~ /(\d+).(\d+).(\d+)/  ;       #Splits
     into $1,$2,and $3'
54 li(7)='$x[4] = join ('+ascii(34)+'-'+ascii(34)+',$1,
     $month[$2],'+ascii(34)+'19$3'+ascii(34)+');'
55 li(8)='$t = (join'+ascii(34)+':'+ascii(34)+', @x);'
56 li(9)='print $t ;'
57
58 li(10)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
     '+ascii(39)+'to go back to console\n\n'+ascii(34)
     +')'
59 halt(' ')
60
61 v=mopen(nam+'.pl','wt')
62 for i=1:9
63     mfprintf(v,"%s\n",li(i))
64     if i~=10 then
65     printf("%s\n",li(i))
66     end
67 end
68 mclose(v)
69 if getos()=='Linux' then
```

```
70      printf("\n\nPlease open a new terminal window
           and then go to the directory %s and execute
           the following instruction\n\nperl %s.pl [
           Command line parameters if any]\n\nThank You
           \n\n",curr,nam)
71      halt(' ')
72      exit
73  end
74
75
76  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
77  printf(" \n       %c %s.pl  emp.lst   %c[ENTER]\n\n"
       ,ascii(34),nam,ascii(34))
78
79  printf("\n$ %s.pl   emp.lst              #to execute
       the perlscript",nam)
80  halt(' ')
81  dos('start')
82  printf("\n\n\n")
83  halt('   ------------------>Executing PerlScript in
       Command Line Prompt<---------------- ')
84  printf("\n\n\n$ exit         #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
85  halt("........# (hit [ENTER] for result)")
86  //clc()
87
88  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
        initial environment')
89  sleep(1000)
90
91  mdelete(nam+'.pl')
92  mdelete('emp.lst')
```

Scilab code Exa 19.16 Program 16

```
1  clear
2  mode(-1)
3  pwd
4  curr=ans
5  clc
6
7  printf("Example 16     :                    Show the
       method of  lowlevel filehandling in perl \n")
8  disp("
       ************************************************************
       ")
9  disp("Answer     :    ")
10
11
12 disp("INSTRUCTIONS    : ")
13 printf("\n1. Here all instructions are preloaded in
       the form of a demo\n\nInitially the whole perl
       script is displaying and then \n the result of
       the same can be seen in the command line
       interpreter.\n\n2. PLEASE MAKE SURE THAT THE
       PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
       THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
       AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
       ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
       \n\n6. Afile named %cemp.lst%c with the necessary
       details gets created automatically for the
       session",ascii(34),ascii(34))
14 halt('.............Press [ENTER] to continue.....')
```

```
15 halt("")
16 clc
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
18
19 i=0
20 i=i+1;f(i)='2233|a.k.shukla                |g.m.
             |sales            |12/12/52|6000'
21 i=i+1;f(i)='9876|jai sharma                |director   |
      production |12/03/50|7000'
22 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m       |
      marketing  |19/04/43|6000'
23 i=i+1;f(i)='2356|barun sengupta       |director   |
      personnel  |11/05/47|7800'
24 i=i+1;f(i)='5423|n.k. gupta               |chairman |
      admin       |30/08/56|5400'
25 i=i+1;f(i)='1006|chanchal singhvi    |director   |
      sales          |03/09/38|6700'
26 i=i+1;f(i)='6213|karuna ganguly       |g.m.       |
      accounts   |05/06/62|6300'
27 i=i+1;f(i)='1265|s.n. dasgupta          |manager  |
      sales          |12/09/63|5600'
28 i=i+1;f(i)='4290|jayant Choudhary  |executive |
      production|07/09/50|6000'
29 i=i+1;f(i)='2476|anil aggarwal         |manager    |
      sales          |01/05/59|5000'
30 i=i+1;f(i)='6521|lalit chowdury        |director     |
      marketing |26/09/45|8200'
31 i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
      accounts   |12/12/55|6000'
32 i=i+1;f(i)='3564|sudhir Agarwal      |executive  |
      personnel  |06/07/47|7500'
33 i=i+1;f(i)='2345|j.b. saxena              |g.m.
               |marketing |12/03/45|8000'
34 i=i+1;f(i)='0110|v.k. agrawal          |g.m.
               |marketing  |31/02/40|9000'
35 n=i
36 printf("\n\n$ cat emp.lst      # to open the file
```

```
    emp.lst")
halt(' ')
u=mopen('emp.lst','wt')
for i=1:n
    mfprintf(u,"%s\n",f(i))
    printf("%s\n",f(i))
end
mclose(u)

printf("\n# Enter the name of the perlscript file
    whichever you desire  \n\n")
nam=input('$ cat  ','s')
halt(' ')
clc
li(1)='#!/usr/bin/perl '
li(2)='# Script: '+nam+'.pl  - Shows use of low-
    level filehandling available in perl  '
li(3)="#"
li(4)='open(FILEIN,'+ascii(34)+'emp.lst'+ascii(34)+'
    ) || die('+ascii(34)+'Cannot open file'+ascii(34)
    +')  ;'
li(5)='open(FILEOUT, '+ascii(34)+'>emp_out.lst'+
    ascii(34)+') ; '
li(6)='while(<FILEIN>) {                          #
    As long as there are lines in the file '
li(7)='             print FILEOUT if (1..3)  ; print
     STDOUT ;    #  Can also use if ($. < 4)  '
li(8)='}'
li(9)='close(FILEIN); '
li(10)='close(FILEOUT); '
li(11)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
    '+ascii(39)+'to go back to console\n\n'+ascii(34)
    +')'
halt(' ')


v=mopen(nam+'.pl','wt')
```

```
65 for i=1:9
66     mfprintf(v,"%s\n",li(i))
67     if i~=10 then
68     printf("%s\n",li(i))
69     end
70 end
71 mclose(v)
72 if getos()=='Linux' then
73     printf("\n\nPlease open a new terminal window
          and then go to the directory %s and execute
          the following instruction\n\nperl %s.pl [
          Command line parameters if any]\n\nThank You
          \n\n",curr,nam)
74     halt(' ')
75     exit
76 end
77
78 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
79 printf(" \n       %c %s.pl     %c[ENTER]\n\n",ascii
     (34),nam,ascii(34))
80
81 printf("\n$ %s.pl              #to execute the
     perlscript",nam)
82 halt(' ')
83 dos('start')
84 printf("\n\n\n")
85 halt('  ------------------>Executing PerlScript in
     Command Line Prompt<---------------- ')
86 printf("\n# Type the following command in command
     prompt as it appears to check the file is
     successfully copied\n\n")
87 printf("\n$ type emp_out.lst              #to check
     the contents of the new file\n')
88 halt(' ')
89 dos('start')
90 printf("\n\n\n")
91 halt('  ------------------>Executing PerlScript in
```

```
    Command Line Prompt<---------------- ')
92  printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
93  halt(".........# (hit [ENTER] for result)")
94  //clc()
95
96  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
97  sleep(1000)
98
99  mdelete(nam+'.pl')
100 mdelete('emp.lst')
101 mdelete('emp_out.lst')
```

Scilab code Exa 19.17 Program 17

```
1  clear
2  mode(-1)
3  pwd
4  curr=ans
5  clc
6  //Program for example 1 chapter 1
7
8  printf("Example 17     :                Show the
    method of  filetesting in perlscript')
9  disp("
    ************************************************************
    ")
10 disp("Answer     :    ")
11
```

```
12 disp("INSTRUCTIONS    : ")
13 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
14 halt('.............Press [ENTER] to continue.....')
15 halt("")
16 clc
17 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
18
19
20 printf("\n# Enter the name of the perlscript file
      whichever you desire  \n\n")
21 nam=input('$ cat  ','s')
22 halt('  ')
23 clc
24 li(1)='#!/usr/bin/perl '
25 li(2)='# Script: '+nam+'.pl  - Find files that are
      less than 2.4 hours old  '
26 li(3)="#"
27 li(4)='foreach $file (`dir /B`) { '
28 li(5)='chop($file) ;'
29 li(6)='if (($m_age = -M $file) < 0.1) {
                        #tenth of a day i.e 2.4 hours
      '
30 li(7)='            printf  '+ascii(34)+' File %s was
      last modified %0.3f day(s) back \n'+ascii(34)+',
      $file,$m_age ; '
31 li(8)='        }'
32 li(9)='}'
33
```

```
34 li(10)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
     '+ascii(39)+'to go back to console\n\n'+ascii(34)
     +')'
35 halt(' ')
36
37 v=mopen(nam+'.pl','wt')
38 for i=1:10
39     mfprintf(v,"%s\n",li(i))
40     if i~=10 then
41     printf("%s\n",li(i))
42     end
43 end
44 mclose(v)
45 if getos()=='Linux' then
46     printf("\n\nPlease open a new terminal window
         and then go to the directory %s and execute
         the following instruction\n\nperl %s.pl [
         Command line parameters if any]\n\nThank You
         \n\n",curr,nam)
47     halt(' ')
48     exit
49 end
50
51 printf("\n# type the following command in the
     command line interpreter as soon as it appears")
52 printf(" \n       %c  %s.pl [THE NUMBERS AS
     COMMANDLINE ARGUMENTS]    %c[ENTER]\n\n",ascii
     (34),nam,ascii(34))
53
54 printf("\n$  %s.pl  [THE NUMBERS AS COMMANDLINE
     ARGUMENTS]             #to execute the perlscript
     ",nam)
55 halt(' ')
56 dos('start')
57 printf("\n\n\n")
58 halt('  ------------------>Executing PerlScript in
     Command Line Prompt<------------------ ')
59 printf("\n\n\n$ exit        #To exit the current
```

```
    simulation terminal and return to Scilab console\
    n\n")
halt(".......# (hit [ENTER] for result)")
//clc()

printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
sleep(1000)

mdelete(nam+'.pl')
```

Scilab code Exa 19.18 Program 18

```
clear
mode(-1)
pwd
curr=ans
clc

printf("Example 18     :                Show the
    method of  declaration of subroutines')
disp("
    ************************************************************
    ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
    the form of a demo\n\nInitially the whole perl
    script is displaying and then \n the result of
    the same can be seen in the command line
```

```
    interpreter.\n\n2. PLEASE MAKE SURE THAT THE
    PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
    THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
    AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
    ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
    \n")
13 halt('.............Press [ENTER] to continue.....')
14 halt("")
15 clc
16 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
17
18
19 printf("\n# Enter the name of the perlscript file
    whichever you desire  \n\n")
20 nam=input('$ cat  ','s')
21 halt(' ')
22 clc
23 li(1)='#!/usr/bin/perl '
24 li(2)='system('+ascii(34)+'cls'+ascii(34)+')   ;#
    Script: '+nam+'.pl  - Shows the use of
    Subroutines  '
25 li(3)="#"
26 li(4)=' system('+ascii(34)+'cls'+ascii(34)+');
                #clears the screen'
27 li(5)='$username = &take_input('+ascii(34)+'Oracle
    user-id: '+ascii(34)+')  ;'
28 li(6)='$password = &take_input('+ascii(34)+'Oracle
    password: '+ascii(34)+', '+ascii(34)+'noecho'+
    ascii(34)+') ;'
29 li(7)='print  '+ascii(34)+'\nThe username and
    password are $username and $password\n'+ascii(34)
    +' ;'
30 li(8)=' '
31 li(9)='sub take_input { '
32 li(10)='       my ($prompt,$flag) = @_ ;
                # @_stores arguments of subroutines
    '
```

```
33 li(11)='        while (1) {
                                                  # (1)
     means always true '
34 li(12)='                    print('+ascii(34)+'$prompt
     '+ascii(34)+') ;'
35 li(13)='                    use Term::ReadKey;'
36 li(14)='                     ReadMode 2        if (@_
     ==2);                            #turn ehoing off'
37 li(15)='                     chop($name=<STDIN>);      '
38 li(16)='                      ReadMode 0      if (@_==2)
     ;                      #turn echoing on back'
39 li(17)='                        last if $name =~ /\w/ ;
        #Quit if $name has atleast one word character'
40 li(18)='                    }'
41 li(19)='     $name    ;          #return $name will
     also do'
42 li(20)='     }'
43
44 li(21)='print('+ascii(34)+'\n\nType'+ascii(39)+'exit
     '+ascii(39)+'to go back to console\n\n'+ascii(34)
     +')'
45 halt(' ')
46
47 v=mopen(nam+'.pl','wt')
48 for i=1:20
49      mfprintf(v,"%s\n",li(i))
50
51      if i==13 then
52          printf('\t  system(%c stty -echo%c) if (@_
              ==2) ;              #Echo off\n',ascii
              (34),ascii(34))
53      end
54       if i==16 then
55          printf('\t  system(%c stty echo%c) if (@_
              ==2) ;              #Echo on\n',ascii(34)
              ,ascii(34))
56      end
57       if i~=20&i~=14&i~=13&i~=16 then
```

```
58      printf("%s\n",li(i))
59      end
60  end
61  mclose(v)
62  if getos()=='Linux' then
63      printf("\n\nPlease open a new terminal window
           and then go to the directory %s and execute
           the following instruction\n\nperl %s.pl [
           Command line parameters if any]\n\nThank You
           \n\n",curr,nam)
64      halt(' ')
65      exit
66  end
67
68  printf("\n# type the following command in the
       command line interpreter as soon as it appears")
69  printf(" \n        %c  %s.pl     %c[ENTER]\n\n",ascii
       (34),nam,ascii(34))
70
71  printf("\n$  %s.pl                #to execute the
       perlscript",nam)
72  halt(' ')
73  dos('start')
74  printf("\n\n\n")
75  halt('   ------------------>Executing PerlScript in
       Command Line Prompt<---------------- ')
76  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
77  halt("........# (hit [ENTER] for result)")
78  //clc()
79
80  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
        initial environment')
81  sleep(1000)
82
83  mdelete(nam+'.pl')
```

Chapter 21

Advanced Shell Programming

Scilab code Exa 21.1 Program 1

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 1     :                    Show the method
       of using arrays in advanced shellscripting \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
```

```
    ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
    \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
16
17 halt('')
18 clc
19 i=0
20 i=i+1;f(i)='#!/usr/bin/ksh'
21 i=i+1;f(i)='# Script: dateval.sh - validates a date
    field using an array'
22 i=i+1;f(i)='IFS='+ascii(34)+'/'+ascii(34)+''
23 i=i+1;f(i)='n='+ascii(34)+'[0-9][0-9]'+ascii(34)+''
24 i=i+1;f(i)='set -A month arr 0 31 29 31 30 31 30 31
    30 31 30 31'
25 i=i+1;f(i)='while echo '+ascii(34)+'Enter a date: \c
    '+ascii(34)+' ; do'
26 i=i+1;f(i)='     read value'
27 i=i+1;f(i)='     case '+ascii(34)+'$value'+ascii(34)
    +' in'
28 i=i+1;f(i)='          '+ascii(34)+''+ascii(34)+')
    echo '+ascii(34)+'No date entered'+ascii(34)+' ;
    continue ;;'
29 i=i+1;f(i)='     $n/$n/$n) set $value'
30 i=i+1;f(i)='               let rem='+ascii(34)+'$3 %
     $4'+ascii(34)+''
31 i=i+1;f(i)='               if [ $2 -gt 12 -o $2 -eq
    0 ] ; then'
32 i=i+1;f(i)='                  echo '+ascii(34)+'
    Illegal month'+ascii(34)+' ; continue'
33 i=i+1;f(i)='               else'
34 i=i+1;f(i)='                  case '+ascii(34)+'
    $value'+ascii(34)+' in'
35 i=i+1;f(i)='               29/02/??) [ $rem -gt 0 ]
    &&'
```

```
36 i=i+1;f(i)='                                { echo '+ascii
   (34)+'20$3 is not a leap year'+ascii(34)+' ;
   continue ; } ;;'
37 i=i+1;f(i)='                             *) [ $1 -gt ${
   month_arr[$2]} -o $1 -eq 0 ] &&'
38 i=i+1;f(i)='                                { echo '+ascii
   (34)+'Illegal day'+ascii(34)+' ; continue ; } ;;'
39 i=i+1;f(i)='                          esac'
40 i=i+1;f(i)='                       fi;;'
41 i=i+1;f(i)='                      *) echo '+ascii(34)+'
   Invalid date'+ascii(34)+' ; continue ;;'
42 i=i+1;f(i)='    esac'
43 i=i+1;f(i)='    echo '+ascii(34)+'$1/$2/$3'+ascii
   (34)+' is a valid date'
44 i=i+1;f(i)='done'
45 n=i
46
47 printf("\n# Enter the name of the shellscript file
   whichever you desire  \n\n")
48 nam=input('$ cat  ','s')
49 halt(' ')
50
51 for i=1:n
52     printf("%s\n",f(i))
53 end
54 halt(' ')
55 clc
56 i=0
57 i=i+1;f(i)='@echo off'
58 i=i+1;f(i)='set chc=y'
59 i=i+1;f(i)=':loop'
60 i=i+1;f(i)='if /I '+ascii(34)+'%chc%'+ascii(34)+'=='
   +ascii(34)+'n'+ascii(34)+' goto endloop'
61 i=i+1;f(i)='set /P dat=Enter a date: '
62 i=i+1;f(i)='if '+ascii(34)+'%dat%'+ascii(34)+' equ '
   +ascii(34)+''+ascii(34)+' echo No date entered&&
   goto chci'
63 i=i+1;f(i)='if exist testt del testt'
```

```
64 i=i+1;f(i)='echo %dat%>testt'
65 i=i+1;f(i)='for /F '+ascii(34)+'tokens=1,2,3 delims
     =/'+ascii(34)+' %%i in (testt) do set dd=%%i&&set
      mm=%%j&&set yy=%%k'
66 i=i+1;f(i)='if %mm% gtr 12 echo Illegal month&&goto
     chci'
67 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'01'+ascii(34)+' set ulim=31&&goto
     printing'
68 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'03'+ascii(34)+' set ulim=31&&goto
     printing'
69 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'04'+ascii(34)+' set ulim=30&&goto
     printing'
70 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'05'+ascii(34)+' set ulim=31&&goto
     printing'
71 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'06'+ascii(34)+' set ulim=30&&goto
     printing'
72 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'07'+ascii(34)+' set ulim=31&&goto
     printing'
73 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'08'+ascii(34)+' set ulim=31&&goto
     printing'
74 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'09'+ascii(34)+' set ulim=30&&goto
     printing'
75 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'10'+ascii(34)+' set ulim=31&&goto
     printing'
76 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'11'+ascii(34)+' set ulim=30&&goto
     printing'
77 i=i+1;f(i)='if '+ascii(34)+'%mm%'+ascii(34)+'=='+
     ascii(34)+'12'+ascii(34)+' set ulim=31&&goto
```

```
    printing'
78  i=i+1;f(i)='set /a rem=yy%%4'
79  i=i+1;f(i)='if %rem% neq 0 set ulim=28&&goto nlpyear
    '
80  i=i+1;f(i)='set ulim=29'
81  i=i+1;f(i)='goto printing'
82  i=i+1;f(i)=':nlpyear'
83  i=i+1;f(i)='if '+ascii(34)+'%dd%'+ascii(34)+'=='+
    ascii(34)+'29'+ascii(34)+' echo 20%yy% is not a
    leap year&&goto chci'
84  i=i+1;f(i)=':printing'
85  i=i+1;f(i)='if %dd% leq %ulim% echo %dat% is a valid
     date&&goto chci'
86  i=i+1;f(i)='echo Illegal day '
87  i=i+1;f(i)=':chci'
88  i=i+1;f(i)='set /p chc=Do you want to continue ? (y/
    n) : '
89  i=i+1;f(i)='goto loop'
90  i=i+1;f(i)=':endloop'
91  i=i+1;f(i)='pause>NUL&&del testt'
92  n=i
93
94  if getos()=='Linux' then
95      printf("\n\nPlease Switch to windows and then
           execute using the instructions\n\nThank You \
           n\n")
96      halt(' ')
97      exit
98  end
99
100 v=mopen(nam+'.sh.bat','wt')
101 for i=1:n
102     mfprintf(v,"%s\n",f(i))
103 end
104 mclose(v)
105
106
107 printf("\n# type the following command in the
```

```
    command line interpreter as soon as it appears")
108 printf(" \n        %c  %s.sh      %c [COMMANDLINE
    ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
109
110 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
111
112 halt(' ')
113 dos('start')
114 printf("\n\n\n")
115 halt('  ------------------>Executing ShellScript in
    Command Line Prompt<---------------- ')
116 printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
117 halt("........# (hit [ENTER] for result)")
118 //clc()
119
120 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
121 sleep(1000)
122
123 mdelete(nam+'.sh.bat')
124 mdelete('emp.lst')
```

Scilab code Exa 21.2 Program 2

```
1 clear
2 flag=1
3 mode(-1)
4 clc
```

```
5
6  printf("Example 2     :                Show the method
       of calling functions in shellscripts \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
16
17 halt('')
18 clc
19 i=0
20 i=i+1;f(i)='#! /bin/sh'
21 i=i+1;f(i)='# Script: user_passwd.sh - Uses a shell
      function'
22 i=i+1;f(i)='#'
23 i=i+1;f(i)='.mainfunc.sh                          #
      Script containing valid_string function'
24 i=i+1;f(i)=''
25 i=i+1;f(i)='valid_string() {'
26 i=i+1;f(i)='    while echo '+ascii(34)+'$1 \c'+ascii
      (34)+' 1>&2 ; do'
```

```
27 i=i+1;f(i)='          read name'
28 i=i+1;f(i)='          case $name in'
29 i=i+1;f(i)='                  '+ascii(34)+''+ascii(34)
     +') echo '+ascii(34)+'Nothing entered'+ascii(34)+
     ' 1>&2 ; continue ;;'
30 i=i+1;f(i)='                   *) if [ `expr '+ascii
     (34)+'$name'+ascii(34)+' : '+ascii(39)+'.*'+ascii
     (39)+'` -gt $2 ] ; then'
31 i=i+1;f(i)='                           echo '+ascii(34)+
     'Maximum $2 characters permitted'+ascii(34)+'
     1>&2'
32 i=i+1;f(i)='                      else'
33 i=i+1;f(i)='                         break'
34 i=i+1;f(i)='                      fi ;;'
35 i=i+1;f(i)='          esac'
36 i=i+1;f(i)='     done'
37 i=i+1;f(i)='     echo $name'
38 i=i+1;f(i)='}'
39 i=i+1;f(i)=''
40 i=i+1;f(i)='user=`valid_string '+ascii(34)+'Enter
     your user-id :'+ascii(34)+' 16`'
41 i=i+1;f(i)='stty -echo                        #
     Password not to be echoed'
42 i=i+1;f(i)='password=`valid_string '+ascii(34)+'
     Enter your password:'+ascii(34)+' 5`'
43 i=i+1;f(i)='stty echo                         # Turns
     on echoing facility'
44 i=i+1;f(i)='echo '+ascii(34)+'\nYour user-id is
     $user and your password is $password'+ascii(34)+'
     '
45 n=i
46 printf("\n# Enter the name of the shellscript file
     whichever you desire  \n\n")
47 nam=input('$ cat  ','s')
48 halt(' ')
49
50 for i=1:n
51     printf("%s\n",f(i))
```

```
end
halt(' ')
clc
i=0
i=i+1;f(i)='#!/usr/bin/perl'
i=i+1;f(i)='$username = &take_input('+ascii(34)+'
   Enter your user-id: '+ascii(34)+', '+ascii(34)+'
   16'+ascii(34)+')  ;'
i=i+1;f(i)='$password = &take_input('+ascii(34)+'
   Enter your password: '+ascii(34)+', '+ascii(34)+'
   5'+ascii(34)+','+ascii(34)+'noecho'+ascii(34)+')
   ;'
i=i+1;f(i)='print  '+ascii(34)+'\nYour user-id is
   $username and your password is $password\n'+ascii
   (34)+' ;'
i=i+1;f(i)='sub take_input { '
i=i+1;f(i)='my ($prompt,$len,$flag) = @_ ;'
i=i+1;f(i)='while (1) { '
i=i+1;f(i)='print('+ascii(34)+'$prompt'+ascii(34)+')
    ;'
i=i+1;f(i)='use Term::ReadKey;'
i=i+1;f(i)='ReadMode 2        if  (@_==3);'
i=i+1;f(i)='chop($name=<STDIN>);     '
i=i+1;f(i)='ReadMode 0      if (@_==3);'
i=i+1;f(i)='if ( length($name) eq 0 ) { printf '+
   ascii(34)+'\nNothing entered\n'+ascii(34)+' ;
   next }'
i=i+1;f(i)='if ( length($name) > $len) { printf '+
   ascii(34)+'Maximum %d characters permitted\n'+
   ascii(34)+' , $len ; next }'
i=i+1;f(i)='last if $name =~ /\w/  ;'
i=i+1;f(i)='}'
i=i+1;f(i)='$name   ;'
i=i+1;f(i)='}'
n=i
```

```
78  v=mopen(nam+'.pl','wt')
79  for i=1:n
80      mfprintf(v,"%s\n",f(i))
81  end
82  mclose(v)
83
84
85  i=0
86  i=i+1;f(i)='@echo off'
87  i=i+1;f(i)=nam+'.pl'
88  i=i+1;f(i)='pause>NUL'
89  i=i+1;f(i)='del  '+nam+'.pl'
90  n=i
91  v=mopen(nam+'.bat','wt')
92  for i=1:n
93      mfprintf(v,"%s\n",f(i))
94  end
95  mclose(v)
96
97  if getos()=='Linux' then
98      printf("\n\nPlease Switch to windows and then
           execute using the instructions\n\nThank You \
           n\n")
99      halt(' ')
100     exit
101 end
102 printf("\n# type the following command in the
       command line interpreter as soon as it appears")
103 printf(" \n       %c %s     %c[ENTER]\n\n",ascii
       (34),nam,ascii(34))
104
105 printf("\n$ %s              #to execute the
       perlscript",nam)
106
107 halt(' ')
108 dos('start')
109 printf("\n\n\n")
110 halt('  ------------------>Executing ShellScript in
```

```
    Command Line Prompt<---------------- ')
printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
halt("........# (hit [ENTER] for result)")
//clc()

printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
sleep(1000)

mdelete(nam+'.bat')
mdelete('emp.lst')
mdelete(nam+'.pl')
```

Scilab code Exa 21.3 Program 3

```
clear
flag=1
mode(-1)
clc

printf("Example 3     :                Show the method
     of using eval in the shellscripts \n")
disp("
    ************************************************************
    ")
disp("Answer     :    ")

disp("INSTRUCTIONS    : ")
printf("\n1. Here all instructions are preloaded in
```

```
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n4. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
12 halt('.............Press [ENTER] to continue.....')
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
16
17 halt('')
18 clc
19
20 printf("\n# Enter the name of the shellscript file
      whichever you desire  \n\n")
21 nam=input('$ cat  ','s')
22 halt(' ')
23 i=0
24 i=i+1;f(i)='#!/bin/sh'
25 i=i+1;f(i)='# Script: dentry2.sh - Uses eval and
      shell functions'
26 i=i+1;f(i)='#'
27 i=i+1;f(i)='trap '+ascii(39)+'echo '+ascii(34)+'
      Program interrupted'+ascii(34)+';exit'+ascii(39)+
      ' HUP INT TERM'
28 i=i+1;f(i)='. mainfunc.sh                    # Invokes
      functions valid_string() and anymore ()'
29 i=i+1;f(i)=''
30 i=i+1;f(i)='prompt1='+ascii(34)+'Employee id : '+
      ascii(34)+' ; prompt2='+ascii(34)+'Name : '+ascii
      (34)+'; prompt3='+ascii(34)+'Designation : '+
      ascii(34)+''
31 i=i+1;f(i)='prompt4='+ascii(34)+'Department : '+
```

```
      ascii(34)+' ; prompt5='+ascii(34)+'Date birth : '
      +ascii(34)+';prompt6='+ascii(34)+'Basic pay : '+
      ascii(34)+''
32  i=i+1;f(i)='rekord='
33  i=i+1;f(i)=''
34  i=i+1;f(i)='flname=`valid_string '+ascii(34)+'Enter
      the output filename: '+ascii(34)+' 8`'
35  i=i+1;f(i)='while true ; do'
36  i=i+1;f(i)='    while [ ${x:=1} -le 6 ] ; do       #
       x first set to 1'
37  i=i+1;f(i)='          eval echo \$prompt$x '+ascii
      (39)+'\\c'+ascii(39)+' 1>&2'
38  i=i+1;f(i)='          read value$x'
39  i=i+1;f(i)='          rekord='+ascii(34)+'${rekord}`
      eval echo \\$value$x`|'+ascii(34)+''
40  i=i+1;f(i)='          x=`expr $x + 1`'
41  i=i+1;f(i)='    done'
42  i=i+1;f(i)='    echo '+ascii(34)+'$rekord'+ascii
      (34)+''
43  i=i+1;f(i)='    anymore '+ascii(34)+'More entries
      to add'+ascii(34)+' 1&>2 || break'
44  i=i+1;f(i)='done > $flname'
45  i=i+1;f(i)=''
46  i=i+1;f(i)='anymore() {'
47  i=i+1;f(i)='    echo '+ascii(34)+'\n$1 ?(y/n) : \c'
      +ascii(34)+' 1>&2'
48  i=i+1;f(i)='    read response'
49  i=i+1;f(i)='    case '+ascii(34)+'$response'+ascii
      (34)+' in'
50  i=i+1;f(i)='        y/Y) echo 1>&2 ; return 0 ;;'
51  i=i+1;f(i)='          *) return 1 ;;'
52  i=i+1;f(i)='    esac'
53  i=i+1;f(i)='}'
54  n=i
55
56  for i=1:n
57      printf("%s\n",f(i))
58  end
```

```
59 halt(' ')
60 clc
61 i=0
62 i=i+1;f(i)='@echo off'
63 i=i+1;f(i)='set /P flname=Enter the output filename:
     '
64 i=i+1;f(i)='if exist %flname% del %flname%'
65 i=i+1;f(i)='set response=y'
66 i=i+1;f(i)=':loop'
67 i=i+1;f(i)='if /I '+ascii(34)+'%response%'+ascii(34)
     +'=='+ascii(34)+'n'+ascii(34)+' goto endloop'
68 i=i+1;f(i)='echo.'
69 i=i+1;f(i)='set /P eid=Employee id : '
70 i=i+1;f(i)='set /P nam=Name : '
71 i=i+1;f(i)='set /P desig=Designation : '
72 i=i+1;f(i)='set /P dept=Department : '
73 i=i+1;f(i)='set /P dob=Date birth : '
74 i=i+1;f(i)='set /P bas=Basic pay : '
75 i=i+1;f(i)='echo.'
76 i=i+1;f(i)='echo %eid%:%nam%:%desig%:%dept%:%dob%:
     %bas%:>>%flname%'
77 i=i+1;f(i)='set /P response=More entries to add ?(y/
     n) : '
78 i=i+1;f(i)='goto loop'
79 i=i+1;f(i)=':endloop'
80 i=i+1;f(i)='pause>NUL'
81 i=i+1;f(i)='echo.'
82 i=i+1;f(i)='echo.'
83 i=i+1;f(i)='set /p res2= Do you want to see the file
     %flname%?(y/n) : '
84 i=i+1;f(i)='if /I '+ascii(34)+'%res2%'+ascii(34)+'==
     '+ascii(34)+'n'+ascii(34)+' goto endd'
85 i=i+1;f(i)='echo $ cat %flname%'
86 i=i+1;f(i)='type %flname%'
87 i=i+1;f(i)=':endd'
88 i=i+1;f(i)='pause>NUL'
89 n=i
90
```

```
91
92  if getos()=='Linux' then
93      printf("\n\nPlease Switch to windows and then
           execute using the instructions\n\nThank You \
           n\n")
94      halt(' ')
95      exit
96  end
97
98
99  v=mopen(nam+'.sh.bat','wt')
100 for i=1:n
101     mfprintf(v,"%s\n",f(i))
102 end
103 mclose(v)
104
105
106 printf("\n# type the following command in the
       command line interpreter as soon as it appears")
107 printf(" \n        %c  %s.sh      %c[ENTER]\n\n",ascii
       (34),nam,ascii(34))
108
109 printf("\n$ %s.sh                #to execute the
       perlscript",nam)
110
111 halt(' ')
112 dos('start')
113 printf("\n\n\n")
114 halt('   ------------------>Executing ShellScript in
       Command Line Prompt<---------------- ')
115 printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
116 halt("........# (hit [ENTER] for result)")
117 //clc()
118
119 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
```

```
120 sleep(1000)
121
122 mdelete(nam+'.sh.bat')
123 mdelete('emp.lst')
```

Scilab code Exa 21.4 Program 4

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 4     :                    Show the method
       of using exec command to make many streams \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9
10 disp("INSTRUCTIONS    : ")
11 printf("\n1. Here all instructions are preloaded in
      the form of a demo\n\nInitially the whole perl
      script is displaying and then \n the result of
      the same can be seen in the command line
      interpreter.\n\n2. PLEASE MAKE SURE THAT THE
      PERLSCRIPT INTERPRETER\nEXISTS IN THE SYSTEM\nOR
      THE COMMAND WOULD NOT WORK \n\n3. PRESS ENTER
      AFTER EACH COMMAND to see its RESULT\n\n5. PRESS
      ENTER AFTER EACH RESULT TO GO TO THE NEXT COMMAND
      \n")
12 halt('.............Press [ENTER] to continue.....')
```

```
13 halt("")
14 clc
15 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
16 i=0
17 i=i+1;f(i)='2233|a.k.shukla                |g.m.
            |sales             |12/12/52|6000'
18 i=i+1;f(i)='9876|jai sharma                |director   |
      production |12/03/50|7000'
19 i=i+1;f(i)='5678|sumit chakrobarty|d.g.m       |
      marketing   |19/04/43|6000'
20 i=i+1;f(i)='2356|barun sengupta       |director   |
      personnel   |11/05/47|7800'
21 i=i+1;f(i)='5423|n.k. gupta                |chairman |
      admin        |30/08/56|5400'
22 i=i+1;f(i)='1006|chanchal singhvi    |director   |
      sales           |03/09/38|6700'
23 i=i+1;f(i)='6213|karuna ganguly       |g.m.         |
      accounts    |05/06/62|6300'
24 i=i+1;f(i)='1265|s.n. dasgupta          |manager   |
      sales           |12/09/63|5600'
25 i=i+1;f(i)='4290|jayant Choudhary   |executive |
      production|07/09/50|6000'
26 i=i+1;f(i)='2476|anil aggarwal         |manager    |
      sales           |01/05/59|5000'
27 i=i+1;f(i)='6521|lalit chowdury        |director     |
      marketing |26/09/45|8200'
28 i=i+1;f(i)='3212|shyam saksena       |d.g.m        |
      accounts    |12/12/55|6000'
29 i=i+1;f(i)='3564|sudhir Agarwal      |executive   |
      personnel   |06/07/47|7500'
30 i=i+1;f(i)='2345|j.b. saxena              |g.m.
               |marketing |12/03/45|8000'
31 i=i+1;f(i)='0110|v.k. agrawal           |g.m.
               |marketing   |31/02/40|9000'
32 n=i
33 printf("\n\n$ cat emp.lst       # to open the file
      emp.lst")
```

```
34 halt(' ')
35 u=mopen('emp.lst','wt')
36 for i=1:n
37     mfprintf(u,"%s\n",f(i))
38     printf("%s\n",f(i))
39 end
40 mclose(u)
41 halt('')
42 clc
43 i=0
44 i=i+1;f(i)='#!/bin/sh'
45 i=i+1;f(i)='# Script: countpat.sh -- Uses exec to
     handle multiple files'
46 i=i+1;f(i)='#'
47 i=i+1;f(i)='exec > $2                  # Open file 1
     for storing selected lines'
48 i=i+1;f(i)='exec 3> $3                 # Open file 3
     for storing patterns not found'
49 i=i+1;f(i)='exec 4> $4                 # Open file 4
     for storing invalid patterns'
50 i=i+1;f(i)=''
51 i=i+1;f(i)='[ $# -ne 4 ] && { echo '+ascii(34)+'4
     arguments required'+ascii(34)+' ; exit 2 ; }'
52 i=i+1;f(i)=''
53 i=i+1;f(i)='exec < $1                   # Redirecting
     output'
54 i=i+1;f(i)='while read pattern ; do '
55 i=i+1;f(i)='      case '+ascii(34)+'$pattern'+ascii
     (34)+' in'
56 i=i+1;f(i)='            ????) grep $pattern emp.lst
      ||'
57 i=i+1;f(i)='                  echo $pattern not
     found in file 1>&3 ;;'
58 i=i+1;f(i)='                *) echo $pattern not a
     four-character string 1>&4 ;;'
59 i=i+1;f(i)='      esac'
60 i=i+1;f(i)='done'
61 i=i+1;f(i)='exec > \/dev/tty        # Redirects
```

```
    standard output back to terminal'
62  i=i+1;f(i)='echo Job Over'
63  n=i
64
65  printf("\n# Enter the name of the shellscript file
    whichever you desire  \n\n")
66  nam=input('$ cat  ','s')
67  halt(' ')
68
69  for i=1:n
70      printf("%s\n",f(i))
71  end
72  halt(' ')
73  clc
74  i=0
75  i=i+1;f(i)='@echo off'
76  i=i+1;f(i)='for %%x in (%*) do set /a ccc+=1'
77  i=i+1;f(i)='if %ccc% neq 4 echo 4 arguments required
    &&goto endd'
78  i=i+1;f(i)='echo.'
79  i=i+1;f(i)='echo ----------Creating file %1
    ----------'
80  i=i+1;f(i)='set chice=y'
81
82  i=i+1;f(i)=':loop1'
83  i=i+1;f(i)='if /I '+ascii(34)+'%chice%'+ascii(34)+'
    =='+ascii(34)+'n'+ascii(34)+' goto endloop1'
84  i=i+1;f(i)='set /P inp=Enter the employee-id : '
85  i=i+1;f(i)='echo %inp%>>%1'
86  i=i+1;f(i)='if exist len del len'
87  i=i+1;f(i)='echo.%inp%>len'
88  i=i+1;f(i)='for /F '+ascii(34)+'usebackq'+ascii(34)+
    ' %%i in ('+ascii(39)+'len'+ascii(39)+') do set
    len=%%~zi'
89  i=i+1;f(i)='del len&&set /a len-=2'
90  i=i+1;f(i)='if %len% neq 4 echo %inp% is not a four-
    character string>>%4&&goto chi'
91  i=i+1;f(i)='if exist res del res'
```

```
92  i=i+1;f(i)='findstr /B '+ascii(34)+'%inp%'+ascii(34)
      +' emp.lst>res'
93  i=i+1;f(i)='for /F '+ascii(34)+'usebackq'+ascii(34)+
      ' %%i in ('+ascii(39)+'res'+ascii(39)+') do set
      siz=%%~zi'
94  i=i+1;f(i)='if %siz% equ 0 echo %inp% not found in
      file>>%3&&goto chi'
95  i=i+1;f(i)='type res>>%2'
96  i=i+1;f(i)=':chi'
97  i=i+1;f(i)='set /P chice=Do you want to continue?(y/
      n) : '
98  i=i+1;f(i)='cls&&goto loop1'
99  i=i+1;f(i)=':endloop1'
100 i=i+1;f(i)='set /P c1=Do you want to see the file %1
      ?(y/n) : '
101 i=i+1;f(i)='if /I '+ascii(34)+'%c1%'+ascii(34)+'=='+
      ascii(34)+'n'+ascii(34)+' goto endloop2'
102 i=i+1;f(i)='type %1'
103 i=i+1;f(i)=':endloop2'
104 i=i+1;f(i)='set /P c2=Do you want to see the file %2
      ?(y/n) : '
105 i=i+1;f(i)='if /I '+ascii(34)+'%c2%'+ascii(34)+'=='+
      ascii(34)+'n'+ascii(34)+' goto endloop3'
106 i=i+1;f(i)='type %2'
107 i=i+1;f(i)=':endloop3'
108 i=i+1;f(i)='set /P c3=Do you want to see the file %3
      ?(y/n) : '
109 i=i+1;f(i)='if /I '+ascii(34)+'%c3%'+ascii(34)+'=='+
      ascii(34)+'n'+ascii(34)+' goto endloop4'
110 i=i+1;f(i)='type %3'
111 i=i+1;f(i)=':endloop4'
112 i=i+1;f(i)='set /P c4=Do you want to see the file %4
      ?(y/n) : '
113 i=i+1;f(i)='if /I '+ascii(34)+'%c4%'+ascii(34)+'=='+
      ascii(34)+'n'+ascii(34)+' goto endloop2'
114 i=i+1;f(i)='type %4'
115 i=i+1;f(i)=':endloop5'
116 i=i+1;f(i)='pause>NUL&&del %1&&del %2&&del %3&&del
```

```
    %4&&&del res'
117 n=i
118
119
120 if getos()=='Linux' then
121     printf("\n\nPlease Switch to windows and then
        execute using the instructions\n\nThank You \
        n\n")
122     halt(' ')
123     exit
124 end
125
126 v=mopen(nam+'.sh.bat','wt')
127 for i=1:n
128     mfprintf(v,"%s\n",f(i))
129 end
130 mclose(v)
131
132
133 printf("\n# type the following command in the
    command line interpreter as soon as it appears")
134 printf(" \n        %c  %s.sh     %c [COMMANDLINE
    ARGUMENTS][ENTER]\n\n",ascii(34),nam,ascii(34))
135
136 printf("\n$ %s.sh  [COMMANDLINE ARGUMENTS]
                #to execute the perlscript",nam)
137
138 halt(' ')
139 dos('start')
140 printf("\n\n\n")
141 halt('  ------------------>Executing ShellScript in
    Command Line Prompt<---------------- ')
142 printf("\n\n\n$ exit         #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
143 halt("........# (hit [ENTER] for result)")
144 //clc()
145
```

```
146 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
        initial environment')
147 sleep(1000)
148
149 mdelete(nam+'.sh.bat')
150 mdelete('emp.lst')
```

Chapter 23

Systems Programming 1 Files

Scilab code Exa 23.1 Program 1

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 1     :                Show the method
       of copying files with the read and write system
      calls \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
      programming in Unix and the commands have NO
      EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp('2.However if possible some selected programmes
       have been TRIED TO BE IMPLEMENTED')
14 halt("")
```

```
15 disp('3.For most of the programmes whose equivalent
     is NOT THERE IN SCILAB,only the output has been
     printed as given in the textbook with no
     interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
     here are also made into a seperate file.If you
     are a unix user then try compiling and running
     the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program ccp.c -- Copies a file with
     the read and write system calls */'
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <fcntl.h>          /* For
     O_RDONLY , O_WRONLY , O_CREAT etc. */'
27 i=i+1;f(i)='#include <sys/stat.h>       /* for
     S_IRUSR , S_IWUSR , S_IRGRP etc. */'
28 i=i+1;f(i)='#define BUFSIZE 1024        /* May not
     be the rigth size here */'
29 i=i+1;f(i)=''
30 i=i+1;f(i)='int main(void) {'
31 i=i+1;f(i)='    int fd1, fd2;           /* File
     descriptors for read and write  */'
32 i=i+1;f(i)='    int n;                  /* Number of
      characters returned by read */'
33 i=i+1;f(i)='    char buf[BUFSIZE];      /* BUFSIZE
     should be carefully chosen */'
34 i=i+1;f(i)='    fd1 = open('+ascii(34)+'/etc/passwd'
     +ascii(34)+',O_RDONLY);'
35 i=i+1;f(i)='    fd2 = open('+ascii(34)+'passwd.bak'+
     ascii(34)+',O_WRONLY | O_CREAT | O_TRUNC ,'
36 i=i+1;f(i)='                S_IRUSR | S_IWUSR |
```

```
      S_IRGRP | S_IWGRP | S_IROTH); /* Mode 664*/'
37 i=i+1;f(i)='     '
38 i=i+1;f(i)='     while ((n = read(fd1, buf, BUFSIZE))
       > 0)   /* Return value of read is */'
39 i=i+1;f(i)='         write(fd2, buf, n);
                          /* used by write as argument
      */'
40 i=i+1;f(i)='         '
41 i=i+1;f(i)='     close(fd1);'
42 i=i+1;f(i)='     close(fd2);'
43 i=i+1;f(i)='     exit(0);                /* This would
       have closed all file descriptors */'
44 i=i+1;f(i)='}'
45 n=i
46 printf("\n\n$ cat ccp.c      # to open the file emp.
      lst")
47 halt(' ')
48 u=mopen('ccp.c','wt')
49 for i=1:n
50     mfprintf(u,"%s\n",f(i))
51     printf("%s\n",f(i))
52 end
53 mclose(u)
54 halt('')
55 clc
56
57 halt(' ')
58  printf("$ cc ccp.c")
59  halt(' ')
60  printf("$ a.out")
61  halt(' ')
62  printf("$ cmp /etc/passwd passwd.bak")
63  halt(' ')
64 if getos()=='Linux' then
65 unix_w('cc ccp.c;a.out;cmp /etc/passwd passwd.bak')
66 else
67  printf("$ _
                                                    #
```

```
    Prompt returns-files identical")
 halt(' ')
end

printf("\n\n\n$ exit          #To exit the current
   simulation terminal and return to Scilab console\
   n\n")
halt(".......# (hit [ENTER] for result)")
//clc()

printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
sleep(1000)
```

Scilab code Exa 23.2 Program 2

```
clear
flag=1
mode(-1)
clc

printf("Example 2     :                Show the method
    of reversing a file using lseek \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
halt(' ')
disp("1.These programs are part of systems
   programming in Unix and the commands have NO
   EQUIVALENT IN SCILAB")
halt(' ')
```

```
13 disp('2.However if possible some selected programmes
       have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
      is NOT THERE IN SCILAB,only the output has been
      printed as given in the textbook with no
      interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
      here are also made into a seperate file.If you
      are a unix user then try compiling and running
      the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
      PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: reverse_read.c -- Reads a
      file in reverse - uses lseek */'
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <fcntl.h>                  /* For
      O_RONLY */'
27 i=i+1;f(i)='#include <unistd.h>                 /* For
      STDOUT_FILENO */'
28 i=i+1;f(i)=''
29 i=i+1;f(i)='int main(int argc, char **argv) {'
30 i=i+1;f(i)='    char buf;                       /*
      Single-character buffer; will make */'
31 i=i+1;f(i)='    int size, fd;                   /* I/O
      inefficient. See Section 23.4 */'
32 i=i+1;f(i)='    '
33 i=i+1;f(i)='    fd= open(argv[1], O_RDONLY);'
34 i=i+1;f(i)='    size = lseek(fd, -1, SEEK_END);  /*
      Pointer taken to EOF - 1 ... */'
35 i=i+1;f(i)='    while (size-- >= 0) {           /*
      ... so siz = file size - 1 */'
```

```
36 i=i+1;f(i)='          read(fd, &buf, 1);          /*
     Read one character at a time */'
37 i=i+1;f(i)='          write(STD_FILENO, &buf, 1); /*
     and write it immediately */'
38 i=i+1;f(i)='          lseek(fd, -2, SEEK_CUR);    /*
     Now move the file pointer back */'
39 i=i+1;f(i)='          }                           /*
     by two characters */'
40 i=i+1;f(i)='        /* exit(0); */              /*
     done deliberately */'
41 i=i+1;f(i)='}'
42 n=i
43 printf("\n\n$ cat reverse_read.c      # to open the
     file emp.lst")
44 halt(' ')
45 u=mopen('reverse_read.c','wt')
46 for i=1:n
47     mfprintf(u,"%s\n",f(i))
48     printf("%s\n",f(i))
49 end
50 mclose(u)
51 halt('')
52 clc
53 printf("\n$ ls    \n")
54 halt(' ')
55 mode(0)
56 ls
57 mode(-1)
58 nam=input("# Please enter a file from the above list
       : ",'s')
59 printf("\n$ cat %s    ",nam)
60 halt(' ')
61 v=mopen(nam,"rt")
62 while ~meof(v)
63     [n,a]=mfscanf(v,"%c");
64     if  meof(v) break
65         end
66     printf("%c",a)
```

```
67 end
68 mclose(v)
69 halt("")
70 printf("\n$ cc reverse_read.c")
71 halt("")
72 printf("$ a.out  %s   \n.....a blank line...
              The terminating \\n of the last line",
   nam)
73 halt("")
74 v=mopen(nam,"rt")
75 mseek(-1,v,'end')
76 siz=mtell(v)
77 siz=siz-1
78 while siz~=-1
79     [n,a]=mfscanf(v,"%c");
80     printf("%c",a)
81     mseek(siz,v)
82     siz=siz-1
83 end
84 mseek(0,v)
85  [n,a]=mfscanf(v,"%c");
86     printf("%c",a)
87 mclose(v)
88 halt(' ')
89
90
91 printf("\n\n\n$ exit         #To exit the current
   simulation terminal and return to Scilab console\
   n\n")
92 halt("........# (hit [ENTER] for result)")
93 //clc()
94
95 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
96 sleep(1000)
```

Scilab code Exa 23.3 Program 3

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 3     :                  Show the effect
       of umask on permissions of a file \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt('  ')
11 disp("1.These programs are part of systems
      programming in Unix and the commands have NO
      EQUIVALENT IN SCILAB")
12 halt('  ')
13 disp('2.However if possible some selected programmes
       have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
      is NOT THERE IN SCILAB,only the output has been
      printed as given in the textbook with no
      interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
      here are also made into a seperate file.If you
      are a unix user then try compiling and running
      the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
```

```
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: umask.c -- Changes umask
     twice and checks effect on permissions */'
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <stdio.h>'
27 i=i+1;f(i)='#include <fcntl.h>'
28 i=i+1;f(i)=''
29 i=i+1;f(i)='int main(void) {'
30 i=i+1;f(i)='    mode_t old_mode,new_mode;'
31 i=i+1;f(i)='    '
32 i=i+1;f(i)='    old_mode = umask(0);
                        /* No mask */'
33 i=i+1;f(i)='    printf('+ascii(34)+'Previous umask
     value: %o\n'+ascii(34)+', old_mode);'
34 i=i+1;f(i)='    '
35 i=i+1;f(i)='    open('+ascii(34)+'foo1'+ascii(34)+',
     O_RDONLY | O_CREAT, 0777);  /* Create file using
     new mask */'
36 i=i+1;f(i)='    umask(old_mode);
                             /* Revert to previous mask
      */'
37 i=i+1;f(i)='    open('+ascii(34)+'foo2'+ascii(34)+',
     O_RDWR | O_CREAT, 0764);    /* Create file using
     old mask */'
38 i=i+1;f(i)='    exit(0);'
39 i=i+1;f(i)='} '
40 n=i
41
42 printf("\n\n$ cat umask.c      # to open the file
     emp.lst")
43 halt(' ')
44 u=mopen('umask.c','wt')
45 for i=1:n
```

```
46      mfprintf(u,"%s\n",f(i))
47      printf("%s\n",f(i))
48  end
49  mclose(u)
50  halt('')
51  clc
52
53  halt(' ')
54   printf("$ cc umask.c")
55   halt(' ')
56   printf("$ a.out")
57   halt(' ')
58   printf("Previous umask value: 22")
59   halt(' ')
60   printf("$ ls -l foo?")
61   halt(' ')
62  disp("-rwxrwxrwx    1  sumit     sumit        0 Dec
        1  12:01    foo1")
63  disp("-rwxr--r--    1  sumit     sumit        0 Dec
        1  12:01    foo2")
64  halt(' ')
65  printf("\n\n\n$ exit        #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
66  halt("........# (hit [ENTER] for result)")
67  //clc()
68
69  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
70  sleep(1000)
```

Scilab code Exa 23.4 Program 4

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 4     :
       Print all the error messages present in the
       system using strerror \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming in Unix and the commands have NO
       EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp('2.However if possible some selected programmes
        have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
       is NOT THERE IN SCILAB,only the output has been
       printed as given in the textbook with no
       interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
       PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: show_errors.c --Uses
```

```
      strerror to print all error messages */'
25  i=i+1;f(i)=''
26  i=i+1;f(i)='#include <stdio.h>'
27  i=i+1;f(i)=''
28  i=i+1;f(i)='int main(void) {'
29  i=i+1;f(i)='    int i;'
30  i=i+1;f(i)='    extern int sys_nerr;          /*
      Total number of error messages */'
31  i=i+1;f(i)='    '
32  i=i+1;f(i)='    for (i=0; i < sys_nerr; i++)'
33  i=i+1;f(i)='         printf('+ascii(34)+'%d: %s\n'+
      ascii(34)+', i, strerror(i));'
34  i=i+1;f(i)='    printf('+ascii(34)+'Number of errors
       available: %d\n'+ascii(34)+', sys_nerr);'
35  i=i+1;f(i)='    exit(0);'
36  i=i+1;f(i)='}'
37  n=i
38
39  printf("\n\n$ cat show_errors.c      # to open the
      file emp.lst")
40  halt(' ')
41  u=mopen('show_errors.c','wt')
42  for i=1:n
43      mfprintf(u,"%s\n",f(i))
44      printf("%s\n",f(i))
45  end
46  mclose(u)
47  halt('')
48  clc
49  halt('')
50  disp('$ cc show_errors.c')
51  halt("")
52  disp("$ a.out")
53  halt("")
54  printf("0: Error 0\n1: Not owner\n2:No such file or
      directory\n3: No such process\n4: Interrupted
      system call\n5: I/O error\n13: Permission denied\
      n")
```

```
55 halt("")
56 printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
57 halt(".......# (hit [ENTER] for result)")
58 //clc()
59
60 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
61 sleep(1000)
```

Scilab code Exa 23.5 Program 5

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 5     :
      Print all the system call errors with perror \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
      programming in Unix and the commands have NO
      EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp('2.However if possible some selected programmes
      have been TRIED TO BE IMPLEMENTED')
14 halt("")
```

```
15 disp('3.For most of the programmes whose equivalent
     is NOT THERE IN SCILAB,only the output has been
     printed as given in the textbook with no
     interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
     here are also made into a seperate file.If you
     are a unix user then try compiling and running
     the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: show_errno.c -- Displaying
     system call errors with perror */'
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <fcntl.h>'
27 i=i+1;f(i)=''
28 i=i+1;f(i)='int main(int argc, char **argv) {'
29 i=i+1;f(i)='    int fd;'
30 i=i+1;f(i)='    char* filename = '+ascii(34)+'
     non_existent_file'+ascii(34)+'; /* This file must
      not exist */'
31 i=i+1;f(i)='    '
32 i=i+1;f(i)='    fd = open(filename, O_RDONLY);
           /* File descriptor assigned first */'
33 i=i+1;f(i)='    if (fd == -1)
                          /* and then checked */'
34 i=i+1;f(i)='        perror('+ascii(34)+'
     no_existent_file'+ascii(34)+');'
35 i=i+1;f(i)='    if ((fd = open('+ascii(34)+'/etc/
     shadow'+ascii(34)+',O_RDONLY)) == -1) /* bOTH
     COMBINED HERE */'
36 i=i+1;f(i)='        perror('+ascii(34)+'shadow'+
     ascii(34)+');'
```

```
37 i=i+1;f(i)='     if ((fd = open('+ascii(34)+'
     show_errno.c'+ascii(34)+',O_WRONLY | O_CREAT |
     O_EXCL, 0744)) == -1)'
38 i=i+1;f(i)='         perror('+ascii(34)+'show_errno.c
     '+ascii(34)+');'
39 i=i+1;f(i)='     exit(0);'
40 i=i+1;f(i)='}'
41 n=i
42 printf("\n\n$ cat show_errorno.c      # to open the
     file emp.lst")
43 halt(' ')
44 u=mopen('show_errorno.c','wt')
45 for i=1:n
46     mfprintf(u,"%s\n",f(i))
47     printf("%s\n",f(i))
48 end
49 mclose(u)
50 halt('')
51 clc
52 halt('')
53 disp('$ cc show_errorno.c')
54 halt("")
55 disp("$ a.out")
56 halt("")
57 printf("non_existent_file: No such file or directory
     \nshadow: Permission denied\nshow_errno.c: File
     exists\n ")
58 halt("")
59 printf("\n\n\n$ exit        #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
60 halt("........# (hit [ENTER] for result)")
61 //clc()
62
63 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
64 sleep(1000)
```

Scilab code Exa 23.6 Program 6

```
clear
flag=1
mode(-1)
clc

printf("Example 6     :                  Show the method
    of reversing a file using error handling
   alternatives \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
halt(' ')
disp("1.These programs are part of systems
   programming in Unix and the commands have NO
   EQUIVALENT IN SCILAB")
halt(' ')
disp('2.However if possible some selected programmes
    have been TRIED TO BE IMPLEMENTED')
halt("")
disp('3.For most of the programmes whose equivalent
   is NOT THERE IN SCILAB,only the output has been
   printed as given in the textbook with no
   interactive input as in the programme below')
halt("")
disp("4.However the .c files which are displayed
   here are also made into a seperate file.If you
   are a unix user then try compiling and running
   the programme with gcc or cc compiler")
```

```
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: reverse_read2.c -- Reads a
     file in reverse - uses error handling */'
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <fcntl.h>              /* For 0
     _RDONLY    */'
27 i=i+1;f(i)='#include <unistd.h>             /* For
     STDOUT_FILENO */'
28 i=i+1;f(i)='#include <errno.h>              /* For
     ENOENT, errno, etc. */'
29 i=i+1;f(i)='#include <stdio.h>              /* For
     ENOENT, errno, etc. */'
30 i=i+1;f(i)=''
31 i=i+1;f(i)='int main(int argc, chr **argv) {'
32 i=i+1;f(i)='    int size, fd;'
33 i=i+1;f(i)='    char buf;                    /* Single
     -character buffer */'
34 i=i+1;f(i)='    char *mesg = '+ascii(34)+'Not enough
     arguments\n'+ascii(34)+' ;'
35  i=i+1;f(i)='    '
36 i=i+1;f(i)='    if (argc != 2) {             /* Our
     own user-defined error message */'
37 i=i+1;f(i)='        write(STDERR_FILENO, mesg,
     strlen(mesg)); /* Crude form of error*/'
38 i=i+1;f(i)='        exit(1);
                                    /*handling using
     write*/'
39 i=i+1;f(i)='    }
                                                /*
     Use fprintf instead*/'
40 i=i+1;f(i)='    '
41 i=i+1;f(i)='    if ((fd = open(argv[1],O_RDONLY)) ==
```

```
      -1) {'
42 i=i+1;f(i)='          if (errno == ENOENT) {
                  /* Checking for specific error*/'
43 i=i+1;f(i)='            fprintf(stderr, '+ascii(34)+
    '%s\n'+ascii(34)+',stderror(errno)); /*perror is
    better*/'
44 i=i+1;f(i)='            exit(2);'
45 i=i+1;f(i)='          } else {'
46 i=i+1;f(i)='            perror(argv[1]);
                /* Using two library functions */'
47 i=i+1;f(i)='            exit(3);
                    /* perror and exit.often
    the  */'
48 i=i+1;f(i)='          }
                                    /* preferred
    way */'
49 i=i+1;f(i)='    }                       '
50 i=i+1;f(i)='    '
51 i=i+1;f(i)='    lseek(fd, 1, SEEK_END);
                /* Pointer taken to EOF + 1 first */
    '
52 i=i+1;f(i)='    while (lseek(fd, -2, SEEK_CUR) >=0)
    { /* and then back by two bytes */    '
53 i=i+1;f(i)='          if (read(fd, &buf, 1) != 1) {
        /* A signal can create error here */'
54 i=i+1;f(i)='            perror('+ascii(34)+'read'+
    ascii(34)+');'
55 i=i+1;f(i)='            exit(4);'
56 i=i+1;f(i)='          }'
57 i=i+1;f(i)='          if (write(STDOUT_FILENO, &buf,
    1) != 1) { /* Disk may run out of space */'
58 i=i+1;f(i)='            perror('+ascii(34)+'write'+
    ascii(34)+');'
59 i=i+1;f(i)='            exit(5);'
60 i=i+1;f(i)='          }'
61 i=i+1;f(i)='    }'
62 i=i+1;f(i)='    close(fd);                /*Can have
    error here too*/'
```

```
63 i=i+1;f(i)='     exit(0);                    /* exit
     doesn'+ascii(39)+'t return - hence no error */'
64 i=i+1;f(i)='}'
65 n=i
66
67 printf("\n\n$ cat reverse_read2.c      # to open the
     file emp.lst")
68 halt(' ')
69 u=mopen('reverse_read2.c','wt')
70 for i=1:n
71     mfprintf(u,"%s\n",f(i))
72     printf("%s\n",f(i))
73 end
74 mclose(u)
75 halt('')
76 clc
77 printf("\n$ ls    \n")
78 halt(' ')
79 mode(0)
80 ls
81 mode(-1)
82 nam=input("# Please enter a file from the above list
     : ",'s')
83 printf("\n$ cat %s    ",nam)
84 halt(' ')
85 v=mopen(nam,"rt")
86 while ~meof(v)
87     [n,a]=mfscanf(v,"%c");
88     if  meof(v) break
89         end
90     printf("%c",a)
91 end
92 mclose(v)
93 halt("")
94 printf("\n$ cc reverse_read2.c")
95 halt("")
96 printf("$ a.out  %s   \n.....a blank line...
              The terminating \\n of the last line",
```

```
    nam)
97  halt("")
98  v=mopen(nam,"rt")
99  mseek(-1,v,'end')
100 siz=mtell(v)
101 siz=siz-1
102 while siz~=-1
103     [n,a]=mfscanf(v,"%c");
104     printf("%c",a)
105     mseek(siz,v)
106     siz=siz-1
107 end
108 mseek(0,v)
109   [n,a]=mfscanf(v,"%c");
110     printf("%c",a)
111
112 mclose(v)
113 halt('  ')
114
115
116 printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
117 halt(".......# (hit [ENTER] for result)")
118 //clc()
119
120 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
121 sleep(1000)
```

Scilab code Exa 23.7 Program 7

```
1 clear
```

```
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 7     :                Show the method
       of directory navigation with chdir and getcwd \n
     ")
7  disp("
     ************************************************************
     ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
     programming in Unix and the commands have NO
     EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp('2.However if possible some selected programmes
      have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
     is NOT THERE IN SCILAB,only the output has been
     printed as given in the textbook with no
     interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
     here are also made into a seperate file.If you
     are a unix user then try compiling and running
     the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: dir.c -- Directory
     navigation with chdir and getcwd */'
```

```
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <stdio.h>'
27 i=i+1;f(i)='#define PATH_LENGTH 200'
28 i=i+1;f(i)=''
29 i=i+1;f(i)='void quit(char *,int );    /* Prototyoe
      definition */'
30 i=i+1;f(i)=''
31 i=i+1;f(i)='int main(int argc, char **argv) {'
32 i=i+1;f(i)='    char olddir[PATH_LENGTH + 1];
      /* Extra character for null */'
33 i=i+1;f(i)='    char newdir[PATH_LENGTH + 1];'
34 i=i+1;f(i)='    '
35 i=i+1;f(i)='    if (getcwd(olddir, PATH_LENGTH) ==
      -1) /* Getting current directory */'
36 i=i+1;f(i)='        quit('+ascii(34)+'getcwd'+ascii
      (34)+', 1);'
37 i=i+1;f(i)='    printf('+ascii(34)+'pwd: %s\n'+ascii
      (34)+',olddir);'
38 i=i+1;f(i)='    '
39 i=i+1;f(i)='    if ((chdir(argv[1]) == -1))        /*
       Changing to another directory */'
40 i=i+1;f(i)='        quit('+ascii(34)+'chdir'+ascii
      (34)+', 2);'
41 i=i+1;f(i)='    printf('+ascii(34)+'cd: %s\n'+ascii
      (34)+', argv[1]);'
42 i=i+1;f(i)='    '
43 i=i+1;f(i)='    getcwd(newdir, PATH_LENGTH);       /*
       getting new directory */'
44 i=i+1;f(i)='    printf('+ascii(34)+'pwd: %s\n'+ascii
      (34)+',newdir);'
45 i=i+1;f(i)='    exit(0);'
46 i=i+1;f(i)='}'
47 n=i
48 printf("\n\n$ cat dir.c      # to open the file emp.
      lst")
49 halt(' ')
50 u=mopen('dir.c','wt')
51 for i=1:n
```

```
52      mfprintf(u,"%s\n",f(i))
53      printf("%s\n",f(i))
54  end
55  mclose(u)
56  halt('')
57  clc
58
59  halt(' ')
60   printf("$ cc dir.c")
61   halt(' ')
62   disp("#  Please enter the name of the directory
        which you want to go as the command line
        argument")
63   disp("")
64  nam=input("$ a.out    ",'s')
65   halt(' ')
66   pwd
67   back=ans
68   printf("pwd:  %s\n",back)
69   printf("cd:  %s\n",nam)
70   printf("pwd: %s
                                                Change of
        directory inside program\n",nam)
71   cd(nam)
72   halt("")
73   printf("$ pwd \n")
74   cd(back)
75   printf("%s                                          ...is
         not available outside it\n",back)
76
77  halt(' ')
78  printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
79  halt("........#  (hit [ENTER] for result)")
80  //clc()
81
82  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
```

```
    initial environment')
83 sleep(1000)
```

Scilab code Exa 23.8 Program 8

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 8     :                Show the method
       of using readdir to populate a dirent structure
      \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt('  ')
11 disp("1.These programs are part of systems
      programming in Unix and the commands have NO
      EQUIVALENT IN SCILAB")
12 halt('  ')
13 disp('2.However if possible some selected programmes
       have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
      is NOT THERE IN SCILAB,only the output has been
      printed as given in the textbook with no
      interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
      here are also made into a seperate file.If you
```

```
      are a unix user then try compiling and running
      the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: lls.c -- Uses readdir to
     populate a dirent structure */'
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <stdio.h>'
27 i=i+1;f(i)='#include <dirent.h>'
28 i=i+1;f(i)=''
29 i=i+1;f(i)='int main(int argc, int **argv) {'
30 i=i+1;f(i)='    DIR *dir;                    /*
     Returned by opendir*/'
31 i=i+1;f(i)='    struct dirent *direntry;    /*
     Returned by readdir*/'
32 i=i+1;f(i)='    '
33 i=i+1;f(i)='    if ((dir = opendir(argv[1])) == NULL
     ) /* Directory must exist and*/'
34 i=i+1;f(i)='        quit('+ascii(34)+'opendir'+ascii
     (34)+', 1);              /* have read permission
     */'
35 i=i+1;f(i)='        '
36 i=i+1;f(i)='    while ((direntry = readdir(dir)) !=
     NULL)   /* Till there are new entries*/'
37 i=i+1;f(i)='           printf('+ascii(34)+'%10d %s\n
     '+ascii(34)+',direntry->d_ino,direntry->d_name);'
38 i=i+1;f(i)='    closedir(dir);'
39 i=i+1;f(i)='    exit(0);'
40 i=i+1;f(i)='}       '
41         n=i
42 printf("\n\n$ cat lls.c      # to open the file emp.
     lst")
43 halt(' ')
```

```
44 u=mopen('lls.c','wt')
45 for i=1:n
46     mfprintf(u,"%s\n",f(i))
47     printf("%s\n",f(i))
48 end
49 mclose(u)
50 halt('')
51 clc
52
53 halt(' ')
54  printf("$ cc dir.c")
55  halt(' ')
56  disp("#  Please enter the name of the directory
       which you want to go as the command line
       argument")
57  disp("")
58 nam=input("$ a.out   ",'s')
59  halt(' ')
60  pwd
61  back=ans
62  cd(nam)
63 x=dir()
64 dt=getdate(x.date);
65 mprintf("%-20s    :    %05d-%03d-%03d              %02d:
     %02d:%02d\n",x.name,dt(:,[1 2 6 7:9]))
66  halt("")
67  cd(back)
68 halt(' ')
69 printf("\n\n\n$ exit          #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
70 halt("........#  (hit [ENTER] for result)")
71 //clc()
72
73 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
74 sleep(1000)
```

Scilab code Exa 23.9 Program 9

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 9     :                Show the method
       of using lstatcall and struct stat to display
      file attributes\n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
      programming in Unix and the commands have NO
      EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp('2.However if possible some selected programmes
       have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
      is NOT THERE IN SCILAB,only the output has been
      printed as given in the textbook with no
      interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
      here are also made into a seperate file.If you
      are a unix user then try compiling and running
      the programme with gcc or cc compiler")
```

```
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: attributes.c -- Uses lstat
     call and struct stat to display file attributes
     */'
25 i=i+1;f(i)='#include <stdio.h>'
26 i=i+1;f(i)='#include <sys/stath>'
27 i=i+1;f(i)=''
28 i=i+1;f(i)='void quit(char *,int);'
29 i=i+1;f(i)=''
30 i=i+1;f(i)='int main(int argc, char**argv) {'
31 i=i+1;f(i)='    struct stat statbuf;     /*We'+ascii
     (39)+'ll use lstat to populate this*/'
32 i=i+1;f(i)='    '
33 i=i+1;f(i)='    if (lstat(argv[1], &statbuf) == -1)'
34 i=i+1;f(i)='       quit('+ascii(34)+'Couldn'+ascii
     (39)+'t stat file'+ascii(34)+', 1);'
35 i=i+1;f(i)='       '
36 i=i+1;f(i)='    printf('+ascii(34)+'File: %s\n'+
     ascii(34)+',argv[1]);'
37 i=i+1;f(i)='    printf('+ascii(34)+'Inode number: %d
     \n'+ascii(34)+',statbuf.st_ino);'
38 i=i+1;f(i)='    printf('+ascii(34)+'UID: %d '+ascii
     (34)+',statbuf.st_uid);'
39 i=i+1;f(i)='    printf('+ascii(34)+'GID: %d \n'+
     ascii(34)+',statbuf.st_gid);'
40 i=i+1;f(i)='    printf('+ascii(34)+'Types and
     Permissions: %o\n'+ascii(34)+',statbuf.st_mode);'
41 i=i+1;f(i)='    printf('+ascii(34)+'Number of links:
     %d \n'+ascii(34)+',statbuf.st_nlink);'
42 i=i+1;f(i)='    printf('+ascii(34)+'Size in bytes:
     %d\n'+ascii(34)+',statbuf.st_size);'
43 i=i+1;f(i)='    printf('+ascii(34)+'Blocks allocated
```

```
      : %d\n'+ascii(34)+',statbuf.st_blocks);'
44  i=i+1;f(i)='     printf('+ascii(34)+'Last
      Modification Time: %s\n'+ascii(34)+',ctime(&
      statbuf.st_mtime));'
45  i=i+1;f(i)='     printf('+ascii(34)+'Last Access Time
      : %s\n'+ascii(34)+',ctime(&statbuf.st_atime));'
46  i=i+1;f(i)='     exit(0);'
47  i=i+1;f(i)='}'
48  n=i
49  printf("\n\n$ cat attributes.c      # to open the
      file emp.lst")
50  halt(' ')
51  u=mopen('attributes.c','wt')
52  for i=1:n
53      mfprintf(u,"%s\n",f(i))
54      printf("%s\n",f(i))
55  end
56  mclose(u)
57  halt('')
58  clc
59
60  halt(' ')
61   printf("$ cc dir.c")
62   halt(' ')
63   disp("")
64  printf("\nHere it displays a mock value since it is
      windows\n\n")
65  if getos()=='Windows' then
66      printf("\n$ a.out /etc/passwd\nFile: /etc/passwd
          \nInode number: 54412\nUID: 0 GID: 3\nType
          and Permissions: 100755\nNumber of links: 1")
67  printf("\nSize in bytes: 10803\nBlocks allocated:
      22\nLast Modification Time: Tue Nov 19 16:29:13
      2002\nLast Access Time: Tue NOv 26 19:57:01 2002\
      n")
68  else
69      printf("$ a.out /etc/passwd")
70      unix_w('cc attributes.c;a.out /etc/passwd')
```

```
71 end
72 halt(' ')
73 printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
74 halt("........# (hit [ENTER] for result)")
75 //clc()
76
77 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
78 sleep(1000)
```

Scilab code Exa 23.10 Program 10

```
1 clear
2 flag=1
3 mode(-1)
4 clc
5
6 printf("Example 10     :                Show the
      method of listing only directories in systems
      programming \n")
7 disp("
      ************************************************************
      ")
8 disp("Answer     :    ")
9 disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
      programming in Unix and the commands have NO
      EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp('2.However if possible some selected programmes
```

```
      have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
     is NOT THERE IN SCILAB,only the output has been
     printed as given in the textbook with no
     interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
     here are also made into a seperate file.If you
     are a unix user then try compiling and running
     the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0;
24 i=i+1;f(i)='/* Program lsdir.c -- Lists only
     directories using S_IFMT and S_ISDR macros */'
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <sys/types.h>'
27 i=i+1;f(i)='#include <sys/stat.h>'
28 i=i+1;f(i)='#include <stdio.h>'
29 i=i+1;f(i)='#include <dirent.h>'
30 i=i+1;f(i)=''
31 i=i+1;f(i)='int main(int argc,int **argv){'
32 i=i+1;f(i)='    DIR *dir;'
33 i=i+1;f(i)='    struct dirent *direntry;      /*
     Returned by readdir() */'
34 i=i+1;f(i)='    struct stat statbuf;          /*
     Address of statbuf used by lstat() */'
35 i=i+1;f(i)='    mode_t file_type, file_perm;'
36 i=i+1;f(i)='    '
37 i=i+1;f(i)='    if ((dir = opendir(argv[1])) == NULL
     )'
38 i=i+1;f(i)='       quit('+ascii(34)+'Couldn'+ascii
     (39)+'t open directory'+ascii(34)+',1);'
```

```
39 i=i+1;f(i)='    if((chdir(argv[1]) == -1)) /* Change
     to directory before */'
40 i=i+1;f(i)='      quit('+ascii(34)+'chdir'+ascii
   (34)+',2);        /* you starting reading its
   entries*/'
41 i=i+1;f(i)='      '
42 i=i+1;f(i)='    while ((direntry = readdir(dir)) !=
   NULL) { /* Read each entry in directory*/'
43 i=i+1;f(i)='      if (lstat(direntry->d_name,&
   statbuf) < 0){ /* dname must be in */'
44 i=i+1;f(i)='        perror('+ascii(34)+'lstat'+
   ascii(34)+');                    /*current
   directory */'
45 i=i+1;f(i)='        continue;'
46 i=i+1;f(i)='        }'
47 i=i+1;f(i)='        if (S_ISDIR(statbuf.st_mode))
   {       /*If file is a directory */'
48 i=i+1;f(i)='          file_type = statbuf.
   st_mode & S_IFMT;'
49 i=i+1;f(i)='          file_perm = statbuf.
   st_mode & ~S_IFMT;'
50 i=i+1;f(i)='          printf('+ascii(34)+'%o %4o
    %s\n'+ascii(34)+', file_type, file_perm,
   direntry->d_name);'
51 i=i+1;f(i)='          }'
52 i=i+1;f(i)='          }'
53 i=i+1;f(i)='          exit(0);'
54 i=i+1;f(i)='}'
55 n=i
56
57 printf("\n\n$ cat lsdir.c      # to open the file
   emp.lst")
58 halt(' ')
59 u=mopen('lsdir.c','wt')
60 for i=1:n
61     mfprintf(u,"%s\n",f(i))
62     printf("%s\n",f(i))
63 end
```

```
64 mclose(u)
65 halt('')
66 clc
67
68 halt(' ')
69  printf("$ cc lsdir.c")
70  halt(' ')
71  printf("\n# Enter the name of the directory as
      command-line argument which you want to access
      \n")
72  nam=input("$ a.out      ",'s')
73  halt(' ')
74  pwd
75  back=ans
76  cd(nam)
77 x=dir()
78 mprintf("40000   755   %s\n",x.name)
79 cd(back)
80  printf("\n\n\n$ exit        #To exit the current
      simulation terminal and return to Scilab console
      \n\n")
81 halt("........# (hit [ENTER] for result)")
82 //clc()
83
84 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
85 sleep(1000)
```

Scilab code Exa 23.11 Program 11

```
1 clear
2 flag=1
3 mode(-1)
```

```
4  clc
5
6  printf("Example 11     :                    Show the
       method of listing all the permissions in a file \
       n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming in Unix and the commands have NO
       EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp('2.However if possible some selected programmes
        have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
       is NOT THERE IN SCILAB,only the output has been
       printed as given in the textbook with no
       interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
       PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: check_all_perm.c -- Checks
       all 12 permission bits of a file */'
25 i=i+1;f(i)=''
26 i=i+1;f(i)='#include <stdio.h>'
```

```
27 i=i+1;f(i)='#include <sys/stat.h>'
28 i=i+1;f(i)='#include <fcntl.h>'
29 i=i+1;f(i)=''
30 i=i+1;f(i)='void print_permissions(char *,struct
     stat *) ;'
31 i=i+1;f(i)='void check_permission(int, int, char *);
     '
32 i=i+1;f(i)=''
33 i=i+1;f(i)='int main(int argc,char *argv[]) {'
34 i=i+1;f(i)='    int i,fd,perm;'
35 i=i+1;f(i)='    char *filename =  argv[1];'
36 i=i+1;f(i)='    struct stat statbuf;'
37 i=i+1;f(i)='    mode_t perm_flag[] = {S_IRUSR,
     S_IWUSR,S_IXUSR,S_IRGRP,S_IWGRP,S_IXGRP,S_IROTH,
     S_IWOTH,S_IXOTH,S_ISUID,S_ISGID,S_ISVTX );'
38 i=i+1;f(i)=''
39 i=i+1;f(i)='    char *mesg[] = {'+ascii(34)+'User-
     readable'+ascii(34)+','+ascii(34)+'User-writable'
     +ascii(34)+','+ascii(34)+'User-executable'+ascii
     (34)+','+ascii(34)+'Group-readable'+ascii(34)+','
     +ascii(34)+'Group-writable'+ascii(34)+','+ascii
     (34)+'Group-executable'+ascii(34)+','+ascii(34)+'
     Others-readable'+ascii(34)+','+ascii(34)+'Others-
     writable'+ascii(34)+','+ascii(34)+'Others-
     executable'+ascii(34)+','+ascii(34)+'SUID bit set
     '+ascii(34)+','+ascii(34)+'SGID bit set'+ascii
     (34)+','+ascii(34)+'Sticky bit set'+ascii(34)+' )
     ;'
40 i=i+1;f(i)=''
41 i=i+1;f(i)='    print_permissions(filename,&statbuf)
     ;'
42 i=i+1;f(i)=' '
43 i=i+1;f(i)='    perm = statbuf.st_mode & -S_IFMT;'
44 i=i+1;f(i)='    for(i = 0; i < 12;i ++)'
45 i=i+1;f(i)='          check_permissions(perm,
     perm_flag[i], mesg[i]);'
46 i=i+1;f(i)='}'
47 n=i
```

```
48
49
50
51 printf("\n\n$ cat check_all_perm.c      # to open
      the file emp.lst")
52 halt(' ')
53 u=mopen('check_all_perm.c','wt')
54 for i=1:n
55     mfprintf(u,"%s\n",f(i))
56     printf("%s\n",f(i))
57 end
58 mclose(u)
59 halt('')
60 clc
61
62 halt(' ')
63  printf("$ cc check_all_perm.c")
64  halt(' ')
65
66  printf("\n$ a.out  /usr/bin/passwd ")
67  halt(' ')
68  printf("\nFile: /usr/bin/passwd     Permissions:
      4511\nUser-readable\nUser-executable\nGroup-
      executable\nOthers-executable\nSUID bit set\n")
69 halt(' ')
70  printf("\n\n\n$ exit        #To exit the current
      simulation terminal and return to Scilab console
      \n\n")
71 halt("........# (hit [ENTER] for result)")
72 //clc()
73
74 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
75 sleep(1000)
```

Scilab code Exa 23.12 Program 12

```
clear
flag=1
mode(-1)
clc

printf("Example 12     :                Show the file
   access rights of a file using the read UID and
   GID \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
halt(' ')
disp("1.These programs are part of systems
   programming in Unix and the commands have NO
   EQUIVALENT IN SCILAB")
halt(' ')
disp('2.However if possible some selected programmes
    have been TRIED TO BE IMPLEMENTED')
halt("")
disp('3.For most of the programmes whose equivalent
   is NOT THERE IN SCILAB,only the output has been
   printed as given in the textbook with no
   interactive input as in the programme below')
halt("")
disp("4.However the .c files which are displayed
   here are also made into a seperate file.If you
   are a unix user then try compiling and running
   the programme with gcc or cc compiler")
```

```
18 disp("5.The inconvenience is regretted.")
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/*Program: faccess.c -- Determines a
     file'+ascii(39)+'s access rigths using read UID
     and GID */'
25 i=i+1;f(i)='#include <stdio.h>'
26 i=i+1;f(i)='#include <unistd.h>'
27 i=i+1;f(i)=''
28 i=i+1;f(i)='void quit(char *,int);'
29 i=i+1;f(i)='int main(int argc, char *argv[]) {'
30 i=i+1;f(i)='    short count;'
31 i=i+1;f(i)='    for (count = 1; count < argc ;++
     count) {'
32 i=i+1;f(i)='        printf('+ascii(34)+'%s  '+ascii
     (34)+',argv[count]);'
33 i=i+1;f(i)='        '
34 i=i+1;f(i)='        if (access(argv[count],F_OK) ==
     -1)'
35 i=i+1;f(i)='        quit('+ascii(34)+'File not found
     '+ascii(34)+',1);'
36 i=i+1;f(i)='        if (access(argv[count],R_OK) ==
     -1)'
37 i=i+1;f(i)='        printf('+ascii(34)+'Not readable
     '+ascii(34)+');'
38 i=i+1;f(i)='        if (access(argv[count],W_OK) ==
     -1)'
39 i=i+1;f(i)='        printf('+ascii(34)+'Not writable
     '+ascii(34)+');'
40 i=i+1;f(i)='        if (access(argv[count],X_OK) ==
     -1)'
41 i=i+1;f(i)='        printf('+ascii(34)+'Not
     executable '+ascii(34)+');'
42 i=i+1;f(i)='        '
```

```
43 i=i+1;f(i)='          printf('+ascii(34)+'\n'+ascii
     (34)+');'
44 i=i+1;f(i)='          }'
45 i=i+1;f(i)='          exit(0);'
46 i=i+1;f(i)='}'
47 n=i
48
49 printf("\n\n$ cat faccess.c      # to open the file
     emp.lst")
50 halt(' ')
51 u=mopen('faccess.c','wt')
52 for i=1:n
53     mfprintf(u,"%s\n",f(i))
54     printf("%s\n",f(i))
55 end
56 mclose(u)
57 halt('')
58 clc
59
60 halt(' ')
61  printf("$ cc faccess.c")
62  halt(' ')
63  printf("$ a.out     /etc/passwd    /etc/shadow")
64  halt(' ')
65 printf("/etc/passwd: Not writable   Not executable\n
     /etc/shadow: Not readable   Not writable   Not
     executable\n\n");
66 halt(' ')
67 printf("\n\n\n$ exit          #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
68 halt("........# (hit [ENTER] for result)")
69 //clc()
70
71 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
72 sleep(1000)
```

Scilab code Exa 23.13 Program 13

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 13     :                Show the
       method of setting a file timestamps \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt('  ')
11 disp("1.These programs are part of systems
       programming in Unix and the commands have NO
       EQUIVALENT IN SCILAB")
12 halt('  ')
13 disp('2.However if possible some selected programmes
        have been TRIED TO BE IMPLEMENTED')
14 halt("")
15 disp('3.For most of the programmes whose equivalent
       is NOT THERE IN SCILAB,only the output has been
       printed as given in the textbook with no
       interactive input as in the programme below')
16 halt("")
17 disp("4.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
18 disp("5.The inconvenience is regretted.")
```

```
19 halt('.............Press [ENTER] to continue.....')
20 halt("")
21 clc
22 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
23 i=0
24 i=i+1;f(i)='/* Program: atimemtime.c -- Sets a file
     time stamps to those of another file */'
25 i=i+1;f(i)='#include <sys/stat.h>'
26 i=i+1;f(i)='#include <fcntl.h>'
27 i=i+1;f(i)='#include <utime.h>'
28 i=i+1;f(i)=''
29 i=i+1;f(i)='void quit(char *,int);'
30 i=i+1;f(i)='int main(int argc,char **argv) {'
31 i=i+1;f(i)='    struct stat statbuf;     /* To
     obtain time stamps for an existing file */'
32 i=i+1;f(i)='    struct utimbuf timebuf;  /* To set
     time stamps for another file */'
33 i=i+1;f(i)='    '
34 i=i+1;f(i)='    if (lstat(argv[1], &statbuf) == -1)'
35 i=i+1;f(i)='        quit('+ascii(34)+'stat'+ascii
     (34)+', 1);'
36 i=i+1;f(i)='        '
37 i=i+1;f(i)='    timebuf.actime = statbuf.st_atime;
     /* Setting members of timebuf with */'
38 i=i+1;f(i)='    timebuf.modtime= statbuf.st_mtime;
     /* values obtained from statbuf */'
39 i=i+1;f(i)='    '
40 i=i+1;f(i)='    if (open(argv[2], O_RWR | O_CREAT,
     0644) == -1)'
41 i=i+1;f(i)='      quit('+ascii(34)+'open'+ascii(34)+
     ', 2);'
42 i=i+1;f(i)='    close(argv[2]);          /*
     Previously used open only to create it */'
43 i=i+1;f(i)='    '
44 i=i+1;f(i)='    if (utime(argv[2]), &timebuf) == -1)
      /* Sets both time stamps for file */'
45 i=i+1;f(i)='      quit('+ascii(34)+'utime'+ascii(34)
```

```
    +', 3);'
46 i=i+1;f(i)='    exit(0);'
47 i=i+1;f(i)='}'
48 n=i
49
50 printf("\n\n$ cat atimemtime.c      # to open the
    file emp.lst")
51 halt(' ')
52 u=mopen('atimemtime.c','wt')
53 for i=1:n
54     mfprintf(u,"%s\n",f(i))
55     printf("%s\n",f(i))
56 end
57 mclose(u)
58 halt('')
59 clc
60
61 halt(' ')
62  printf("$ cc atimemtime.c")
63  halt(' ')
64  printf("$ mv a.out $HOME;cd;a.out .profile .
    logintime")
65  halt(' ')
66  printf("$ ls -l .logintime ; ls -lu .logintime")
67  halt(' ')
68  printf("-rw-r--r--  1 <user> <group>  0 Jun 20
    00:55 .logintime\n")
69  printf("-rw-r--r--  1 <user> <group>  0 Jun  5
    00:30 .logintime\n")
70 halt(' ')
71
72 printf("\n\n\n$ exit        #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
73 halt("........# (hit [ENTER] for result)")
74 //clc()
75
76 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
```

```
        initial environment')
77  sleep(1000)
```

Chapter 24

Systems Programming 2 Process Control

Scilab code Exa 24.1 Program 1

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 1     :                Show the method
       of showing all the type of process IDs \n")
7  disp("
      ************************************************************
      ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
      programming PURELY in Unix and the commands have
      NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
      here are also made into a seperate file.If you
```

```
      are a unix user then try compiling and running
      the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
     OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: process.c -- Lists process
     and user credentials The PID, PPID, real and
     effective UIDs and GIDs */'
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='int main(void) {'
26 i=i+1;f(i)='    printf('+ascii(34)+'PID : %4d,PPDI :
     %4d\n'+ascii(34)+',getpid(),getppid());'
27 i=i+1;f(i)='    printf('+ascii(34)+'UID : %4d, GID :
     %4d\n'+ascii(34)+',getuid(),getgid());'
28 i=i+1;f(i)='    printf('+ascii(34)+'EUID : %4d,EGID
     : %4d\n'+ascii(34)+',geteuid(),getegid());'
29 i=i+1;f(i)='    exit(0);'
30 i=i+1;f(i)='}'
31 n=i
32
33 printf("\n\n$ cat process.c      # to open the file
     emp.lst")
34 halt(' ')
35 u=mopen('process.c','wt')
36 for i=1:n
37     mfprintf(u,"%s\n",f(i))
38     printf("%s\n",f(i))
39 end
40 mclose(u)
41 halt('')
```

```
42 clc
43
44 halt(' ')
45  printf("$ cc process.c")
46  halt(' ')
47  printf("$ a.out")
48  halt(' ')
49  printf("PID : 1035, PPID: 1028\nUID :  102,  GID:
      10\nEUID: 102, EGID:  10\n")
50  halt(' ')
51
52 printf("\n\n\n$ exit          #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
53 halt("........# (hit [ENTER] for result)")
54 //clc()
55
56 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
57 sleep(1000)
```

Scilab code Exa 24.2 Program 2

```
1 clear
2 flag=1
3 mode(-1)
4 clc
5
6 printf("Example 2     :                Show the method
     of showing PID,PPID in both parent and child
    process \n")
7 disp("
    ************************************************************
```

```
    ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
halt(' ')
disp("1.These programs are part of systems
    programming PURELY in Unix and the commands have
    NO EQUIVALENT IN SCILAB")
halt(' ')
disp("2.However the .c files which are displayed
    here are also made into a seperate file.If you
    are a unix user then try compiling and running
    the programme with gcc or cc compiler")
halt(' ')
disp("3.The outputs displayed here are just MOCK
    OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
halt(' ')
disp("4.The inconvenience is regretted.")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
i=0
i=i+1;f(i)='/* Program: fork.c -- A simple fork
    Shows PID,PPID in both parent and child*/'
i=i+1;f(i)=''
i=i+1;f(i)='#include <stdio.h>'
i=i+1;f(i)='#include <sys/types.h>'
i=i+1;f(i)=''
i=i+1;f(i)='int main(void) {'
i=i+1;f(i)='    pid_t pid;'
i=i+1;f(i)='    '
i=i+1;f(i)='    printf('+ascii(34)+'Before forking\n
    '+ascii(34)+');'
i=i+1;f(i)='    pid = fork();          /* Replicates
    current processes */'
i=i+1;f(i)='    '
i=i+1;f(i)='    if(pid >0) {           /* In the
```

```
      parent process; make sure */'
35 i=i+1;f(i)='     sleep(1);              /* That the
      parent does not die before child */'
36 i=i+1;f(i)='     printf('+ascii(34)+'PARENT -- PID:
     %d PPID %d, CHILD PID: %d\n'+ascii(34)+',getpid()
      ,getppid(),pid);}'
37 i=i+1;f(i)='     '
38 i=i+1;f(i)='     else if (pid == 0)     /* In the
      child process */'
39 i=i+1;f(i)='          printf('+ascii(34)+'CHILD --
      PID: %d PPID: %d\n'+ascii(34)+',getpid(),getppid
      ());'
40 i=i+1;f(i)='     else {                  /* pid must
      be -1 here */'
41 i=i+1;f(i)='          printf('+ascii(34)+'Fork error\
     n'+ascii(34)+');'
42 i=i+1;f(i)='          exit(1);}'
43 i=i+1;f(i)='     printf('+ascii(34)+'Both process
      continue from here\n'+ascii(34)+');  /* In both
      processes */'
44 i=i+1;f(i)='     exit(0);'
45 i=i+1;f(i)='}'
46 n=i
47
48 printf("\n\n$ cat fork.c      # to open the file emp
     .lst")
49 halt(' ')
50 u=mopen('fork.c','wt')
51 for i=1:n
52     mfprintf(u,"%s\n",f(i))
53     printf("%s\n",f(i))
54 end
55 mclose(u)
56 halt('')
57 clc
58
59 halt(' ')
60  printf("$ cc fork.c")
```

```
61  halt(' ')
62  printf("$ a.out")
63  halt(' ')
64  printf("Before forking\nCHILD -- PID: 1556 PPID:
       1555\nBoth processes continue from here
       # This statement runs in child\nPARENT -- PID:
       1555 PPID: 1450,CHILD PID: 1556\nBoth processes
       continue from here          ...as well as in
       parent\n")
65  halt(' ')
66
67 printf("\n\n\n$ exit          #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
68 halt("........# (hit [ENTER] for result)")
69 //clc()
70
71 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
72 sleep(1000)
```

Scilab code Exa 24.3 Program 3

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 3     :     Show the effect of
      changing the childs environment and check its
      effect in parent \n")
7  disp("
      ************************************************************
```

```
    ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
halt(' ')
disp("1.These programs are part of systems
    programming PURELY in Unix and the commands have
    NO EQUIVALENT IN SCILAB")
halt(' ')
disp("2.However the .c files which are displayed
    here are also made into a seperate file.If you
    are a unix user then try compiling and running
    the programme with gcc or cc compiler")
halt(' ')
disp("3.The outputs displayed here are just MOCK
    OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
halt(' ')
disp("4.The inconvenience is regretted.")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
i=0
i=i+1;f(i)='/* Program: childenv.c -- Changes child'
    +ascii(39)+'s environment and then checks the
    effect in parent*/'
i=i+1;f(i)='#include <stdio.h>'
i=i+1;f(i)='#include <sys/types.h>'
i=i+1;f(i)='#define PATH_LENGTH 30'
i=i+1;f(i)=''
i=i+1;f(i)='int main(void) {'
i=i+1;f(i)='    pid_t pid;'
i=i+1;f(i)='    int x = 100;'
i=i+1;f(i)='    char newdir[PATH_LENGTH + 1]; /*
    Additional space required for \0*/'
i=i+1;f(i)='    '
i=i+1;f(i)='    getcwd(newdir, PATH_LENGTH);  /* Get
     current directory before fork */'
```

```
34 i=i+1;f(i)='     printf('+ascii(34)+'BEFORE FORK --
     Current directory: %s\n'+ascii(34)+', newdir);'
35 i=i+1;f(i)='     '
36 i=i+1;f(i)='     pid = fork ();'
37 i=i+1;f(i)='     switch (pid) {'
38 i=i+1;f(i)='          case -1:'
39 i=i+1;f(i)='          perror('+ascii(34)+'fork'+ascii
     (34)+');'
40 i=i+1;f(i)='          exit(1);                    /*for
     error*/'
41 i=i+1;f(i)='          case 0:  /*Child*/'
42 i=i+1;f(i)='          printf('+ascii(34)+'CHILD --
     Inherited value of x: %d\n'+ascii(34)+', x);'
43 i=i+1;f(i)='          x=200;'
44 i=i+1;f(i)='          printf('+ascii(34)+'CHILD --
     Changed value of x: %d\n'+ascii(34)+', x);'
45 i=i+1;f(i)='          printf('+ascii(34)+'CHILD --
     Inherited value of PATH: %s\n'+ascii(34)+',
     getenv('+ascii(34)+'PATH'+ascii(34)+'));'
46 i=i+1;f(i)='          setenv('+ascii(34)+'PATH'+ascii
     (34)+','+ascii(34)+'.'+ascii(34)+', 1);   /*
     Change PATH here; use putenv('+ascii(34)+'PATH=.'
     +ascii(34)+') */'
47 i=i+1;f(i)='                                      /* if
     setenv() not supported */'
48 i=i+1;f(i)='          printf('+ascii(34)+'CHILD -- New
      value of PATH: %s\n'+ascii(34)+', getenv('+ascii
     (34)+'PATH'+ascii(34)+'));'
49 i=i+1;f(i)='          if (chdir('+ascii(34)+'/etc'+
     ascii(34)+') != -) {        /* '+ascii(34)+'cd'+
     ascii(34)+' to /etc */'
50 i=i+1;f(i)='          getcwd(newdir, PATH_LENGTH);
     /* Do a '+ascii(34)+'pwd'+ascii(34)+' */'
51 i=i+1;f(i)='          printf('+ascii(34)+'CHILD --
     Current directory changed to: %s\n'+ascii(34)+',
     newdir);'
52 i=i+1;f(i)='          }'
53 i=i+1;f(i)='          break;'
```

```
54 i=i+1;f(i)='          exit(0);'
55 i=i+1;f(i)='          default:/* Parent */'
56 i=i+1;f(i)='          sleep(2);
      /* Allow child to complete */'
57 i=i+1;f(i)='          getcwd(newdir, PATH_LENGTH);
      /*Getting new directory */'
58 i=i+1;f(i)='          printf('+ascii(34)+'PARENT --
      Value of x after change by child: %d\n'+ascii(34)
      +',x);'
59 i=i+1;f(i)='          printf('+ascii(34)+'PARENT --
      Current directory is still: %s\n'+ascii(34)+',
      newdir);'
60 i=i+1;f(i)='          printf('+ascii(34)+'PARENT --
      Value of PATH is unchanged: %s\n'+ascii(34)+',
      getenv('+ascii(34)+'PATH'+ascii(34)+'));'
61 i=i+1;f(i)='          exit(0);'
62 i=i+1;f(i)='}'
63 i=i+1;f(i)='}'
64 n=i
65
66
67 printf("\n\n$ cat childenv.c      # to open the file
       emp.lst")
68 halt(' ')
69 u=mopen('childenv.c','wt')
70 for i=1:n
71     mfprintf(u,"%s\n",f(i))
72     printf("%s\n",f(i))
73 end
74 mclose(u)
75 halt('')
76 clc
77
78 halt(' ')
79  printf("$ cc childenv.c")
80  halt(' ')
81  printf("$ a.out")
82  halt(' ')
```

```
83  printf("BEFORE FORK -- Current directory: /users1/
      home/staff/sumit")
84  printf("\nCHILD -- Inherited value of x: 100")
85  printf("\nCHILD -- Changed value of x:200")
86  printf("\nCHILD -- Inherited value of PATH: /usr/
      bin::/usr/local/bin:/usr/ccs/bin")
87  printf("\nCHILD -- New value of PATH: .")
88  printf("\nCHILD -- Current directory changed to: /
      etc")
89  printf("\nPARENT -- Value of x after change to
      child: 100")
90  printf("\nPARENT -- Current directory is still: /
      users1/home/staff/sumit")
91  printf("\nPARENT -- Value of PATH is unchanged: /
      usr/bin::/usr/local/bin::/usr/ccs/bin")
92  halt(' ')
93
94 printf("\n\n\n$ exit          #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
95 halt(".......# (hit [ENTER] for result)")
96 //clc()
97
98 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
99 sleep(1000)
```

Scilab code Exa 24.4 Program 4

```
1 clear
2 flag=1
3 mode(-1)
4
```

```
5  printf("Example 4     :     Show the effect of
       obtaining child termination status by WEXITSTATUS
        \n")
6  disp("
       ************************************************************
       ")
7  disp("Answer      :    ")
8  disp("INSTRUCTIONS    : ")
9  halt(' ')
10 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
11 halt(' ')
12 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
13 halt(' ')
14 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
15 halt(' ')
16 disp("4.The inconvenience is regretted.")
17 halt('.............Press [ENTER] to continue.....')
18 halt("")
19 clc
20 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
       PRELOADED COMMANDS)\n\n\n")
21 i=0
22 i=i+1;f(i)='/* Program: wait.c -- Uses wait to
       obtain child'+ascii(39)+'s termination status.The
        WEXITSTATUS macro fetches the exit status */'
23 i=i+1;f(i)='#include <stdio.h>'
24 i=i+1;f(i)='#include <fcntl.h>'
25 i=i+1;f(i)='#include <sys/wait.h>'
26 i=i+1;f(i)=''
27 i=i+1;f(i)='int main(int argc, char **argv) {'
28 i=i+1;f(i)='    int fd,exitstatus;'
29 i=i+1;f(i)='    int exitval = 10;   /* Value to be
```

```
      returned by child */'
30 i=i+1;f(i)='     '
31 i=i+1;f(i)='     fd= open(argv[1], O_WRONLY | O_CREAT
      | O_TRUNC, 0644);'
32 i=i+1;f(i)='     write(fd, '+ascii(34)+'Original
      process writes\n'+ascii(34)+',24); /* First write
       */'
33      i=i+1;f(i)=''
34 i=i+1;f(i)='     switch(fork()) {'
35 i=i+1;f(i)='        case 0:'
36 i=i+1;f(i)='             write(fd,'+ascii(34)+'Child
      writes\n'+ascii(34)+',13);        /* Second write
      */'
37 i=i+1;f(i)='             close(fd);         /*Closing
      here doesn'+ascii(39)+'t affect parent'+ascii(39)
      +'s copy */'
38 i=i+1;f(i)='             printf('+ascii(34)+'CHILD:
      Terminating with exit value %d\n'+ascii(34)+',
      exitval);'
39 i=i+1;f(i)='             exit(exitval);    /* Can also
       use_exit(exitval) */'
40 i=i+1;f(i)='             '
41 i=i+1;f(i)='             default:'
42 i=i+1;f(i)='                     wait(&exitstatus);
      /* Waits for child to die */'
43 i=i+1;f(i)='                     printf('+ascii(34)+'
      PARENT: Child terminated with exit value %d\n'+
      ascii(34)+',WEXITSTATUS(exitstatus));'
44 i=i+1;f(i)='

      /*Extracting exit status */'
45 i=i+1;f(i)='                     write(fd, '+ascii
      (34)+'Parent writes\n'+ascii(34)+', 14);       /*
      Third write */'
46 i=i+1;f(i)='                     exit(20);
      /* Value returned to shell; try echo $? */'
47 i=i+1;f(i)='                     }'
48 i=i+1;f(i)='}'
```

```
49 n=i
50
51
52 printf("\n\n$ cat wait.c      # to open the file ")
53 halt(' ')
54 u=mopen('wait.c','wt')
55 for i=1:n
56     mfprintf(u,"%s\n",f(i))
57     printf("%s\n",f(i))
58 end
59 mclose(u)
60 halt('')
61 clc
62
63 halt(' ')
64  printf("$ cc wait.c")
65  halt(' ')
66  printf("$ a.out  foo")
67  halt(' ')
68  printf("CHILD: Terminating with exit value 10\
      nPARENT: Child terminated with exit value 10\n")
69  halt(' ')
70  printf("$ cat foo ")
71  halt(' ')
72  printf("Original process writes\nChild writes\
      nParent writes\n")
73  halt(' ')
74
75 printf("\n\n\n$ exit        #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
76 halt("........# (hit [ENTER] for result)")
77 //clc()
78
79 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
80 sleep(1000)
```

Scilab code Exa 24.5 Program 5

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 5     :     Show the effect of
       creating an orphan by letting child sleep for 2
       minutes where parent dies immediately \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
```

```
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: orphan.c -- Creates an
     orphan by letting child sleep for 2 minutes.'
24 i=i+1;f(i)='                              Parent doesn'+
     ascii(39)+'t call wait and dies immediately */'
25 i=i+1;f(i)='#include <stdio.h>'
26 i=i+1;f(i)='int main(void) {'
27 i=i+1;f(i)='    int pid;'
28 i=i+1;f(i)='    if ((pid = fork()) > 0)      /*
     Parent */'
29 i=i+1;f(i)='    exit(10);                    /*
     Parent exits without calling wait */'
30 i=i+1;f(i)='    else if (pid == 0) {         /* Child
      */'
31 i=i+1;f(i)='    sleep(2);                    /* Lets
     parent die in this time frame */'
32 i=i+1;f(i)='    printf('+ascii(34)+'CHILD: Adopted
     by init now, PPID: %d\n'+ascii(34)+',getppid());'
33 i=i+1;f(i)='    exit(0);'
34 i=i+1;f(i)='}'
35 i=i+1;f(i)='}'
36 n=i
37
38
39 printf("\n\n$ cat orphan.c      # to open the file
     emp.lst")
40 halt(' ')
41 u=mopen('orphan.c','wt')
42 for i=1:n
43     mfprintf(u,"%s\n",f(i))
44     printf("%s\n",f(i))
45 end
46 mclose(u)
47 halt('')
48 clc
49
```

```
50 halt(' ')
51  printf("$ cc orphan.c")
52  halt(' ')
53  printf("$ a.out                          ....no
      response for 2 seconds...")
54  halt(' ')
55  sleep(2000)
56  printf("CHILD: Adopted by init now, PPID: 1")
57  halt(' ')
58  printf("$ echo $?")
59  halt(' ')
60  printf("10")
61  halt(' ')
62
63 printf("\n\n\n$ exit         #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
64 halt("........# (hit [ENTER] for result)")
65 //clc()
66
67 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
68 sleep(1000)
```

Scilab code Exa 24.6 Program 6

```
1 clear
2 flag=1
3 mode(-1)
4 clc
5
6 printf("Example 6     :     Show the effect of using
     the execl series \n")
```

```
 7 disp("
     ************************************************************
     ")
 8 disp("Answer     :    ")
 9 disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
     programming PURELY in Unix and the commands have
     NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
     here are also made into a seperate file.If you
     are a unix user then try compiling and running
     the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
     OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: execl.c -- Uses execl to run
      wc */'
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='int main(void) {'
26 i=i+1;f(i)='    execl('+ascii(34)+'/bin/wc'+ascii
     (34)+','+ascii(34)+'wc'+ascii(34)+','+ascii(34)+'
     -l'+ascii(34)+','+ascii(34)+'-c'+ascii(34)+','+
     ascii(34)+'/etc/passwd'+ascii(34)+',(char*) 0);'
27 i=i+1;f(i)='    printf('+ascii(34)+'execl error\n'+
     ascii(34)+');'
28 i=i+1;f(i)='}'
29 n=i
30
```

```
31
32 printf("\n\n$ cat execl.c      # to open the file
      emp.lst")
33 halt(' ')
34 u=mopen('execl.c','wt')
35 for i=1:n
36     mfprintf(u,"%s\n",f(i))
37     printf("%s\n",f(i))
38 end
39 mclose(u)
40 halt('')
41 clc
42
43 halt(' ')
44  printf("$ cc execl.c")
45  halt(' ')
46  printf("$ a.out                          ")
47  halt(' ')
48  printf("            166       9953    /etc/passwd")
49  halt(' ')
50
51 printf("\n\n\n$ exit         #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
52 halt("........# (hit [ENTER] for result)")
53 //clc()
54
55 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
56 sleep(1000)
```

Scilab code Exa 24.7 Program 7

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 7     :     Show the effect of using
       the execv series \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer      :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
       PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: execv.c -- Stuffs all
       command line arguments to an an array to be used
       with execv */'
24 i=i+1;f(i)=''
25 i=i+1;f(i)='#include <stdio.h>'
26 i=i+1;f(i)='int main(int argc, char **argv) {'
```

```
27 i=i+1;f(i)='     char *cmdargs[] = { '+ascii(34)+'
     grep'+ascii(34)+', '+ascii(34)+'-i'+ascii(34)+',
     '+ascii(34)+'-n'+ascii(34)+', '+ascii(34)+'SUMIT'
     +ascii(34)+', '+ascii(34)+'/etc/passwd'+ascii(34)
     +', NULL };'
28 i=i+1;f(i)='     execv('+ascii(34)+'/bin/grep'+ascii
     (34)+', cmdargc);          /* Execute another
     program*/'
29 i=i+1;f(i)='     printf('+ascii(34)+'execv error\n'+
     ascii(34)+');'
30 i=i+1;f(i)='}'
31 n=i;
32
33
34
35 printf("\n\n$ cat execv.c      # to open the file
     emp.lst")
36 halt(' ')
37 u=mopen('execv.c','wt')
38 for i=1:n
39     mfprintf(u,"%s\n",f(i))
40     printf("%s\n",f(i))
41 end
42 mclose(u)
43 halt('')
44 clc
45
46 halt(' ')
47  printf("$ cc execv.c")
48  halt(' ')
49  printf("$ a.out                          ")
50  halt(' ')
51  printf("15:sumit:x:102:10::/users1/home/staff/sumit
      :/usr/bin/bash")
52  halt(' ')
53
54 printf("\n\n\n$ exit         #To exit the current
     simulation terminal and return to Scilab console\
```

```
    n\n")
55 halt("........# (hit [ENTER] for result)")
56 //clc()
57
58 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
59 sleep(1000)
```

Scilab code Exa 24.8 Program 8

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 8     :     Show the effect of using
     the exec command to run a unix command \n")
7  disp("
     ************************************************************
     ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
     programming PURELY in Unix and the commands have
     NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
     here are also made into a seperate file.If you
     are a unix user then try compiling and running
     the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
```

```
    OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
    PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: exec_and_fork.c -- Uses fork
    , exec and wait to run a unix command */'
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='#include <wait.h>'
26 i=i+1;f(i)=''
27 i=i+1;f(i)='int main(int argc,char **argv) {'
28 i=i+1;f(i)='    int returnval;          /* Used by
    wait*/'
29 i=i+1;f(i)='    '
30 i=i+1;f(i)='    switch(fork()) {'
31 i=i+1;f(i)='          case 0:           /* Run
    command in child */'
32 i=i+1;f(i)='          if ((exec(argv[1], &argv[2]) <
     0 )) {'
33 i=i+1;f(i)='              fprintf(stderr, '+ascii
    (34)+'execl error\n'+ascii(34)+');'
34 i=i+1;f(i)='              exit(200);'
35 i=i+1;f(i)='              }'
36 i=i+1;f(i)='          default:              /* In
    the parent */'
37 i=i+1;f(i)='            wait(&returnval);  /* After
     the command has completed .. */'
38 i=i+1;f(i)='            fprintf(stderr,'+ascii(34)+
    'Exit status: %d\n'+ascii(34)+',WEXITSTATUS(
    returnval));'
39 i=i+1;f(i)='            exit(0);'
40 i=i+1;f(i)='            }'
41 i=i+1;f(i)='}'
42 n=i
```

```
43
44
45
46 printf("\n\n$ cat exec_and_fork.c      # to open the
       file emp.lst")
47 halt(' ')
48 u=mopen('exec_and_fork.c','wt')
49 for i=1:n
50     mfprintf(u,"%s\n",f(i))
51     printf("%s\n",f(i))
52 end
53 mclose(u)
54 halt('')
55 clc
56
57 halt(' ')
58  printf("$ cc exec_and_fork.c")
59  halt(' ')
60  printf("$ a.out   /bin/grep   grep   -i -n  SUMIT
       /etc/passwd                    ")
61  halt(' ')
62  printf("15:sumit:x:102:10::/users1/home/staff/sumit
      :/usr/bin/bash\nExit status: 0")
63  halt(' ')
64
65 printf("\n\n\n$ exit        #To exit the current
      simulation terminal and return to Scilab console\
      n\n")
66 halt("........# (hit [ENTER] for result)")
67 //clc()
68
69 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
70 sleep(1000)
```

Scilab code Exa 24.9 Program 9

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 9     :     Show the effect of
       creating a mock child shell which accepts and
       executes commands \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
```

```
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: shell.c -- Accepts user
     input as a command to be executed. Users the
     strtok library function for parsing command line
     */'
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='#include <unistd.h>'
26 i=i+1;f(i)='#include <string.h>        /*for strtok
     */'
27 i=i+1;f(i)='#include <wait.h>'
28 i=i+1;f(i)=''
29 i=i+1;f(i)='#define BUFSIZE 200         /*Maximum
     size for the command line */'
30 i=i+1;f(i)='#define ARGVSIZE 40         /* Maximum
     number of arguments*/'
31 i=i+1;f(i)='#define DELIM '+ascii(34)+'\n\t\r'+ascii
     (34)+'       /* White-space delimiters for strtok
      */'
32 i=i+1;f(i)=''
33 i=i+1;f(i)='int main(int argc, char **argv) {'
34 i=i+1;f(i)='    int i,n ;'
35 i=i+1;f(i)='    char buf[BUFSIZE + 1];       /*
     Stores the entered command line */'
36 i=i+1;f(i)='    char *clargs[ARGVSIZE];      /*
     Stores the arguments strings */'
37 i=i+1;f(i)='    int returnval ;              /* Used
     by wait */'
38 i=i+1;f(i)='    for (;;) {                   /* Loop
     forever */'
39 i=i+1;f(i)='        n = 1;'
40 i=i+1;f(i)='        write(STDOUT_FILENO, '+ascii
     (34)+'Shell> '+ascii(34)+',7);  /*Display a
     prompt */'
41 i=i+1;f(i)='        read(STDIN_FILENO, buf, BUFSIZE
     );    /* Read user input into buf */'
42 i=i+1;f(i)='        if (!strcmp(buf, '+ascii(34)+'
```

```
      exit\n'+ascii(34)+'))'
43 i=i+1;f(i)='          exit(0);                    /*
      Terminate if user enters exit */'
44 i=i+1;f(i)='                                      /* Now
       parse buf to extract the */'
45 i=i+1;f(i)='          clargs[0] = strtok(buf, DELIM);
       /* first word */'
46 i=i+1;f(i)='                                       /*
      Continue parsing until ... */'
47 i=i+1;f(i)='          while ((clargs[n] = strtok(NULL
      , DELIM) != NULL)'
48 i=i+1;f(i)='           n++;                       /*
      ...all words are extracted */'
49 i=i+1;f(i)='          '
50 i=i+1;f(i)='          clargs[n] = NULL;           /*
      Set last arguments pointer to NULL */'
51 i=i+1;f(i)='          switch(fork()) {'
52 i=i+1;f(i)='                              case 0:'
53 i=i+1;f(i)='                                   if ((execvp
      (clargs[0], &clargs[0])) < 0)'
54 i=i+1;f(i)='                                   exit(200);
         /* We will check this value later */'
55 i=i+1;f(i)='                              default:       /*
      Int the parent */'
56 i=i+1;f(i)='                              wait(&returnval);
      /*After the command has completed.. */'
57 i=i+1;f(i)='                              printf('+ascii(34)+
      'Exit status of command: %d\n'+ascii(34)+',
      WEXITSTATUS(returnval));'
58 i=i+1;f(i)='                              for (i=0; i<=n ;i
      ++)  /*...initialise both... */'
59 i=i+1;f(i)='                              clargs[i] = '+ascii
      (34)+'\0'+ascii(34)+';      /*the argument array
      ...*/'
60 i=i+1;f(i)='                              for (i=0; i<BUFSIZE
      +1; i++)'
61 i=i+1;f(i)='                              buf[i] = '+ascii
      (39)+'\0'+ascii(39)+';      /* ... and the buffer
```

```
      that stores command */'
62  i=i+1;f(i)='                          }
      /* line, so next command can work with */'
63  i=i+1;f(i)='              }
      /* an initialized buffer and argument */'
64  i=i+1;f(i)='}
                                                /* array
      */'
65  i=i+1;f(i)=''
66  n=i
67
68
69
70  printf("\n\n$ cat shell.c       # to open the file
      emp.lst")
71  halt(' ')
72  u=mopen('shell.c','wt')
73  for i=1:n
74      mfprintf(u,"%s\n",f(i))
75      printf("%s\n",f(i))
76  end
77  mclose(u)
78  halt('')
79  clc
80
81  halt(' ')
82   printf("$ cc shell.c")
83   halt(' ')
84   printf("$ a.out ")
85   halt(' ')
86   printf("Shell> ")
87   sleep(1500)
88   printf("grep  joker  /etc/passwd")
89   halt(' ')
90   printf("Exit status of command: 1
                          #grep returns 1 if pattern
       not found")
91   halt(' ')
```

```
92  printf("Shell> ")
93  sleep(1500)
94  printf("pwd                                    #Is this
       the shell builtin? ")
95 halt(' ')
96 printf("/users1/home/staff/sumit\nExit status of
     command: 0")
97  halt(' ')
98  printf("Shell> ")
99 sleep(1500)
100 printf("ls -lu  /usr/bin/pwd                  # Now
     check the access time of on-disk pwd")
101 halt(' ')
102 printf("-r-xr-xr-x      1  root         bin      4360
      May  29  01:33     /use/bin/pwd\nExit status of
     command: 0         # Disk file has just been
     accessed! ")
103  halt(' ')
104  printf("Shell> ")
105 sleep(1500)
106 printf("exit")
107 halt('')
108 printf("$_                # Back to parent shell")
109 halt(' ')
110
111 printf("\n\n\n$ exit          #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
112 halt("........# (hit [ENTER] for result)")
113 //clc()
114
115 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
      initial environment')
116 sleep(1000)
```

Scilab code Exa 24.10 Program 10

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 10     :     Show the effect of using
       dup command to achieve both input and output
       redirection\n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
```

```
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: dup.c -- Uses dup to achieve
     both input and output redirection Closes
     standard streams first before using dup */'
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='#include <unistd.h>'
26 i=i+1;f(i)='#include <sys/stat.h>'
27 i=i+1;f(i)='#include <fcntl.h>'
28 i=i+1;f(i)='#define MODE600 (S_IRUSR | S_IWUSR)'
29 i=i+1;f(i)=''
30 i=i+1;f(i)='int main(int argc,char **argv) {'
31 i=i+1;f(i)='    int fd1,int fd2;'
32 i=i+1;f(i)='    fd1 = open(argv[1],O_RDONLY);'
33 i=i+1;f(i)='    fd2 = open(argv[2], O_WRONLY |
     O_CREAT |O_TRUNC, MODE600);'
34 i=i+1;f(i)='    '
35 i=i+1;f(i)='    close(STDIN_FILENO);    /*This
     should return descriptor 0 */'
36 i=i+1;f(i)='    dup(fd1);'
37 i=i+1;f(i)='    close(STDOUT_FILENO);    /*This
     should return descriptor 1 */'
38 i=i+1;f(i)='    dup(fd2);'
39 i=i+1;f(i)='    '
40 i=i+1;f(i)='    execvp(argv[3], &argv[3]);    /*
     Execute any filter */'
41 i=i+1;f(i)='    printf('+ascii(34)+'Failed to exec
     filter'+ascii(34)+');'
42 i=i+1;f(i)='}'
43 n=i
44
45
46 printf("\n\n$ cat dup.c      # to open the file emp.
     lst")
47 halt(' ')
48 u=mopen('dup.c','wt')
49 for i=1:n
```

```
50      mfprintf(u,"%s\n",f(i))
51      printf("%s\n",f(i))
52  end
53  mclose(u)
54  halt('')
55  clc
56
57  halt(' ')
58   printf("$ cc shell.c")
59   halt(' ')
60   printf("$ a.out   /etc/passwd   passwd.cnt   wc   -1"
       )
61   halt(' ')
62   printf("cat passwd.cnt")
63  halt(' ')
64  printf("         37    /etc/passwd")
65  halt(' ')
66  printf("\n\n\n$ exit          #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
67  halt("........# (hit [ENTER] for result)")
68  //clc()
69
70  printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
71  sleep(1000)
```

Scilab code Exa 24.11 Program 11

```
1  clear
2  flag=1
3  mode(-1)
4  clc
```

```
5
6  printf("Example 11     :     Show the effect of
       opening files int the parent and the child to
       reassign descriptors \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
       PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: dup2.c -- Opens files in the
        parent and uses dup2 in the child to reassign
       the descriptors */'
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='#include <unistd.h>'
26 i=i+1;f(i)='#include <sys/stat.h>'
27 i=i+1;f(i)='#include <fcntl.h>'
28 i=i+1;f(i)='#include <wait.h>'
29 i=i+1;f(i)=''
```

```
30 i=i+1;f(i)='#define OPENFLAGS (O_WRONLY | O_CREAT |
     O_TRUNC)'
31 i=i+1;f(i)='#define MODE600 (S_IRUSR | S_IWUSR)'
32 i=i+1;f(i)=''
33 i=i+1;f(i)='void quit(char *message, int exit_status
     );'
34 i=i+1;f(i)=''
35 i=i+1;f(i)='int main(int argc,char **argv) {'
36 i=i+1;f(i)='    int fd1, fd2, rv, exit_status;'
37 i=i+1;f(i)='    '
38 i=i+1;f(i)='    if (fork() == 0) {        /* Child */
     '
39 i=i+1;f(i)='    if ((fd1 = pen(argv[1], O_RDONLY))
     == -1)'
40 i=i+1;f(i)='    quit('+ascii(34)+'Error in opening
     file for reading\n'+ascii(34)+', 1);'
41 i=i+1;f(i)='    if ((fd2 = open(argv[2], OPENFLAGS,
     MODE600) == -1)'
42 i=i+1;f(i)='    quit('+ascii(34)+'Error in opening
     file for writing\n'+ascii(34)+',1);'
43 i=i+1;f(i)='    dup(fd1,0);          /* Closes
     standard input simultaneously */'
44 i=i+1;f(i)='    dup(fd2,1);          /* Closes
     standard output simultaneously*/'
45 i=i+1;f(i)='    execvp(argv[3], &argv[3]);  /*
     Execute command */'
46 i=i+1;f(i)='    quit('+ascii(34)+'exec error'+ascii
     (34)+', 2);'
47 i=i+1;f(i)='} else {          /*parent */'
48 i=i+1;f(i)='wait(&rv);        /* Or use waitpid(-1,&
     rv, 0) */'
49 i=i+1;f(i)='printf('+ascii(34)+'Exit status: %d\n'+
     ascii(34)+',WEXITSTATUS(rv));'
50 i=i+1;f(i)='}'
51 i=i+1;f(i)='}'
52 n=i
53
54
```

```
55
56 printf("\n\n$ cat dup2.c      # to open the file emp
     .lst")
57 halt(' ')
58 u=mopen('dup2.c','wt')
59 for i=1:n
60     mfprintf(u,"%s\n",f(i))
61     printf("%s\n",f(i))
62 end
63 mclose(u)
64 halt('')
65 clc
66
67 halt(' ')
68  printf("$ cc shell.c")
69  halt(' ')
70  printf("$ a.out /etc/passwd passwd.cnt grep joker")
71  halt(' ')
72  printf("Exit status: 1                              #
      joker not found in /etc/passwd")
73  halt("a.out /etc/passwd    passwd.cnt   grep sumit"
     )
74 printf("Exit status: 0                              #
     sumit found in /etc/passwd ")
75 halt(' ')
76 printf("$ cat passwd.cnt")
77 halt(' ')
78 printf("sumit:x:500:500:sumitabha das:/home/sumit:/
     bin/bash")
79 halt(' ')
80 printf("\n\n\n$ exit        #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
81 halt("........# (hit [ENTER] for result)")
82 //clc()
83
84 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
```

```
85 sleep(1000)
```

Scilab code Exa 24.12 Program 12

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 12     :     Show the effect of
       sharing a pipe between two processes from parent
       to child \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
```

```
20 clc
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: pipe.c -- Shares a pipe
     between two processes for data to flow from
     parent to child*/'
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='#include <unistd.h>'
26 i=i+1;f(i)=''
27 i=i+1;f(i)='void quit(char *, int);'
28 i=i+1;f(i)='int main(void) {'
29 i=i+1;f(i)='    int n,fd[2];        /* fd[2] to be
     filled up by pipe() */'
30 i=i+1;f(i)='    char buf[100];      /* Buffer to be
     used by read() */'
31 i=i+1;f(i)='    '
32 i=i+1;f(i)='    if (pipe(fd) < 0)  /* fd[0] is read
     end */'
33 i=i+1;f(i)='    quit('+ascii(34)+'pipe'+ascii(34)+'
     ,1);    /* fd[1] is write end */'
34  i=i+1;f(i)='    '
35 i=i+1;f(i)='    switch (fork()) {    /* Pipe has
     four descriptors now */'
36 i=i+1;f(i)='    case -1:quit('+ascii(34)+'Fork error
     '+ascii(34)+',2);'
37 i=i+1;f(i)='     case 0:close(fd[1]); /* CHILD-Close
      write end of pipe */'
38 i=i+1;f(i)='            n=read(fd[0],buf,100);  /*
     and read from its read end */'
39 i=i+1;f(i)='            write(STDOUT_FILENO,buf,n);'
40 i=i+1;f(i)='            break;'
41 i=i+1;f(i)='     default: close(fd[0]);  /*PARENT-
     Close read end of pipe */'
42 i=i+1;f(i)='            write(fd[1],'+ascii(34)+'
     Writing to pipe\n'+ascii(34)+', 16); /* write to
     write end */'
43 i=i+1;f(i)='    }'
```

```
44 i=i+1;f(i)='      exit(0);'
45 i=i+1;f(i)='}'
46 n=i
47
48
49 printf("\n\n$ cat pipe.c      # to open the file emp
     .lst")
50 halt(' ')
51 u=mopen('pipe.c','wt')
52 for i=1:n
53     mfprintf(u,"%s\n",f(i))
54     printf("%s\n",f(i))
55 end
56 mclose(u)
57 halt('')
58 clc
59
60 halt(' ')
61  printf("$ cc shell.c")
62  halt(' ')
63  printf("$ a.out ")
64  halt(' ')
65  printf("Writing to pipe")
66
67 halt(' ')
68 printf("\n\n\n$ exit        #To exit the current
     simulation terminal and return to Scilab console\
     n\n")
69 halt("........# (hit [ENTER] for result)")
70 //clc()
71
72 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
     initial environment')
73 sleep(1000)
```

Scilab code Exa 24.13 Program 13

```
clear
flag=1
mode(-1)
clc

printf("Example 13     :     Show the method of
   running two programs in a pileline \n")
disp("
   ************************************************************
   ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
halt(' ')
disp("1.These programs are part of systems
   programming PURELY in Unix and the commands have
   NO EQUIVALENT IN SCILAB")
halt(' ')
disp("2.However the .c files which are displayed
   here are also made into a seperate file.If you
   are a unix user then try compiling and running
   the programme with gcc or cc compiler")
halt(' ')
disp("3.The outputs displayed here are just MOCK
   OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
halt(' ')
disp("4.The inconvenience is regretted.")
halt('.............Press [ENTER] to continue.....')
halt("")
clc
printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
```

```
    PRELOADED COMMANDS)\n\n\n")
22  i=0
23  i=i+1;f(i)='/* Program pipe2.c -- Runs two programs
    in a pipeline Child runs cat, parent runs tr */'
24  i=i+1;f(i)='#include <stdio.h>'
25  i=i+1;f(i)='#include <unistd.h>'
26  i=i+1;f(i)=''
27  i=i+1;f(i)='void quit(char *message,int exit_status)
    ;'
28  i=i+1;f(i)='int main(void) { '
29  i=i+1;f(i)='    int fd[2];              /* To be
    fille dup by pipe() */'
30  i=i+1;f(i)='    '
31  i=i+1;f(i)='    if (pipe(fd) < 0)       /* Now have
    four descriptors for pipe */'
32  i=i+1;f(i)='        quit('+ascii(34)+'pipe'+ascii
    (34)+', 1);'
33  i=i+1;f(i)='    switch (fork()) {'
34  i=i+1;f(i)='            case -1: quit('+ascii(34)+'
    fork'+ascii(34)+', 2);'
35  i=i+1;f(i)='            '
36  i=i+1;f(i)='            case 0: close(fd[0]); /*
    CHILD - Close read end first */'
37  i=i+1;f(i)='                    dup2(fd[1],
    STDOUT_FILENO); /* Connect stdout to write end */
    '
38  i=i+1;f(i)='                    close(fd[1]);       /*
     and close original descriptor */'
39   i=i+1;f(i)='                   execlp('+ascii(34)+'
     cat'+ascii(34)+', '+ascii(34)+'cat'+ascii(34)+',
     '+ascii(34)+'/etc/hosts.equiv'+ascii(34)+', (
     char *) 0);'
40  i=i+1;f(i)='                    '
41  i=i+1;f(i)='            default: close(fd[1]);      /*
     PARENT -- Close write end first */'
42  i=i+1;f(i)='                    dup2(fd[0],
    STDIN_FILENO);    /* Connect stdin to read end */'
43  i=i+1;f(i)='                    close(fd[0]);
```

```
            /* and close original descriptor */'
44  i=i+1;f(i)='                        execlp('+ascii(34)+'
       tr'+ascii(34)+', '+ascii(34)+'tr'+ascii(34)+', '+
       ascii(34)+''+ascii(39)+'[a-z]'+ascii(39)+''+ascii
       (34)+','+ascii(34)+''+ascii(39)+'[A-Z]'+ascii(39)
       +''+ascii(34)+',(char *) 0);'
45  i=i+1;f(i)='                        quit('+ascii(34)+'tr
       '+ascii(34)+', 4);'
46  i=i+1;f(i)='}'
47  i=i+1;f(i)='}'
48  n=i
49
50
51  printf("\n\n$ cat pipe2.c      # to open the file
       emp.lst")
52  halt(' ')
53  u=mopen('pipe2.c','wt')
54  for i=1:n
55      mfprintf(u,"%s\n",f(i))
56      printf("%s\n",f(i))
57  end
58  mclose(u)
59  halt('')
60  clc
61
62  halt(' ')
63   printf("$ cc shell.c")
64   halt(' ')
65   printf("$ a.out ")
66   halt(' ')
67   printf("SATURN\nEARTH\nMERCURY\nJUPITER\n")
68
69  halt(' ')
70  printf("\n\n\n$ exit         #To exit the current
       simulation terminal and return to Scilab console\
       n\n")
71  halt("........# (hit [ENTER] for result)")
72  //clc()
```

```
73
74 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
       initial environment')
75 sleep(1000)
```

Scilab code Exa 24.14 Program 14

```
1  clear
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 14     :  Show the use of generating
       signals and how the system can be made to catch
       it  \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
```

```
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: signal.c -- Waits for 5
     seconds for user input and then generates SIGALRM
      that has a handler specified */'
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='#include <unistd.h>'
26 i=i+1;f(i)='#include <signal.h>'
27 i=i+1;f(i)='#define BUFSIZE 100'
28 i=i+1;f(i)=''
29 i=i+1;f(i)='void alrm_handler(int signo);    /*
     Prototypes declarations for */'
30 i=i+1;f(i)='void quit(char *message,int exit_status)
     ;  /*signal handler and quit */'
31 i=i+1;f(i)=''
32 i=i+1;f(i)='char buf[BUFSIZE] = '+ascii(34)+'foo\0'+
     ascii(34)+';          /* Global variable */'
33 i=i+1;f(i)='int main(void) {'
34 i=i+1;f(i)='    int n;'
35 i=i+1;f(i)='    if (signal(SIGALRM, alrm_handler) ==
      SIG_ERR)   /* signal returns SIG_ERR */'
36 i=i+1;f(i)='    quit('+ascii(34)+'sigalrm'+ascii(34)
     +',1);                              /*on error*/'
37 i=i+1;f(i)='    '
38 i=i+1;f(i)='    fprintf(stderr, '+ascii(34)+'Enter
     filename: '+ascii(34)+');'
39 i=i+1;f(i)='    alarm(5);                          /*
     Sets alarm clock;will deliver */'
40 i=i+1;f(i)='    n = read(STDIN_FILENO, buf, BUFSIZE)
     ; /* SIGALRM in 5 seconds */'
41 i=i+1;f(i)='    if (n>1)
                                    /* Will come here
     if user inputs */'
```

```
42 i=i+1;f(i)='          printf('+ascii(34)+'Filename:
     %s\n'+ascii(34)+',buf);      /* string within 5
     seconds */'
43 i=i+1;f(i)='     exit(0);'
44 i=i+1;f(i)='}'
45 i=i+1;f(i)=''
46 i=i+1;f(i)='void alrm_handler(int signo) {       /*
     Invoked with process recieves SIGALRM */'
47 i=i+1;f(i)='          signal(SIGALRM, alrm_handler);
     /* Resetting signal handler */'
48 i=i+1;f(i)='          fprintf(stderr, '+ascii(34)+'\
     nSignal %d reeived, default filename: %s\n'+ascii
     (34)+', signo, buf);'
49 i=i+1;f(i)='          exit(1);'
50 i=i+1;f(i)='}'
51 n=i
52
53
54 printf("\n\n$ cat signal.c      # to open the file
     emp.lst")
55 halt(' ')
56 u=mopen('signal.c','wt')
57 for i=1:n
58     mfprintf(u,"%s\n",f(i))
59     printf("%s\n",f(i))
60 end
61 mclose(u)
62 halt('')
63 clc
64
65 halt(' ')
66  printf("$ cc signal.c")
67  halt(' ')
68  printf("$ a.out ")
69  halt(' ')
70 printf("Enter filename:  ")
71 sleep(1000);printf("s");sleep(300);printf("i");sleep
     (300);printf("g") ;sleep(300);printf("n") ;sleep
```

```
    (300);printf("a") ;sleep(300);printf("l") ;sleep
    (300);printf(".") ;sleep(300);printf("l") ;sleep
    (300);printf("o") ;sleep(300);printf("g") ;sleep
    (500) ;printf(" [-ENTER-]")
printf("\nFilename: signal.log")
halt('')
printf("$ a.out")
halt(' ')
printf("Enter filename:
   # Do not enter anything")
sleep(5000)
printf("# Nothing entered in 5 seconds \n")
printf("Signal 14 received, default filename: foo")
halt(' ')
printf("$ kill -l | grep 14                    #
   What is signal 14")
halt(' ')
printf("13) SIGPIPE      14) SIGALRM      15)
   SIGTERM        16) SIGUSRI\n")
halt(' ')
printf("\n\n\n$ exit        #To exit the current
   simulation terminal and return to Scilab console\
   n\n")
halt("........# (hit [ENTER] for result)")
//clc()

printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
sleep(1000)
```

Scilab code Exa 24.15 Program 15

```
clear
```

```
2  flag=1
3  mode(-1)
4  clc
5
6  printf("Example 15     :        Show the effect of [
       Ctrl-c] in the shell so as to do some operations
       \n")
7  disp("
       ************************************************************
       ")
8  disp("Answer     :    ")
9  disp("INSTRUCTIONS    : ")
10 halt(' ')
11 disp("1.These programs are part of systems
       programming PURELY in Unix and the commands have
       NO EQUIVALENT IN SCILAB")
12 halt(' ')
13 disp("2.However the .c files which are displayed
       here are also made into a seperate file.If you
       are a unix user then try compiling and running
       the programme with gcc or cc compiler")
14 halt(' ')
15 disp("3.The outputs displayed here are just MOCK
       OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
16 halt(' ')
17 disp("4.The inconvenience is regretted.")
18 halt('.............Press [ENTER] to continue.....')
19 halt("")
20 clc
21 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
       PRELOADED COMMANDS)\n\n\n")
22 i=0
23 i=i+1;f(i)='/* Program: signal2.c -- Handles SIGINT
       and SIGTSTP generated from terminal Required two
       [Ctrl-c]s to terminate */'+ascii(39)+''
24 i=i+1;f(i)='#include <stdio.h>'
25 i=i+1;f(i)='#include <unistd.h>'
26 i=i+1;f(i)='#include <signal.h>'
```

```
27 i=i+1;f(i)''
28 i=i+1;f(i)'void quit(char *message, int exit_status
     );'
29 i=i+1;f(i)'void tstp_handler(int signo);     /*
     Handler for [Ctrl-z] */'
30 i=i+1;f(i)'void int_handler(int signo);      /*
     Handler for [Ctrl-c] */'
31 i=i+1;f(i)'int n, count=0;'
32 i=i+1;f(i)''
33 i=i+1;f(i)'int main(void) {'
34 i=i+1;f(i)'    signal(SIGTSTP, tstp_handler); /*
     Disposition for these two signals */'
35 i=i+1;f(i)'    signal(SIGINT, int_handler);    /*
     set to enter respective handler */'
36 i=i+1;f(i)'    signal(SIGQUIT, SIG_IGN);       /*
     Disposition set to ignore */'
37 i=i+1;f(i)'    '
38 i=i+1;f(i)'    fprintf(stderr,'+ascii(34)+'Press [
     Ctrl-z] first, then [Ctrl-c]\n'+ascii(34)+');'
39 i=i+1;f(i)'    for(;;)'
40 i=i+1;f(i)'     pause();                    /* Will
     return on reciept of help */'
41 i=i+1;f(i)'}'
42 i=i+1;f(i)''
43 i=i+1;f(i)'void tstp_handler(int signo) {'
44 i=i+1;f(i)'     signal(SIGTSTP, tstp_handler);
           /* Not entirely reliable */'
45 i=i+1;f(i)'     fprintf(stderr, '+ascii(34)+'Can'+
     ascii(39)+'t stop this program\n'+ascii(34)+');
     /* same signal can reset */'
46 i=i+1;f(i)'}
                                                /*
     disposition to default */'
47 i=i+1;f(i)''
48 i=i+1;f(i)'void int_handler(int signo) {
                /* Will terminate program */'
49 i=i+1;f(i)'       signal(SIGINT, int_handler);
           /* on second invocation */'
```

```
50 i=i+1;f(i)='        (++count == 1) ? printf('+ascii
     (34)+'Press again\n'+ascii(34)+') : quit('+ascii
     (34)+'Quitting'+ascii(34)+', 1);'
51 n=i
52
53
54
55 printf("\n\n$ cat signal2.c      # to open the file
     emp.lst")
56 halt(' ')
57 u=mopen('signal2.c','wt')
58 for i=1:n
59     mfprintf(u,"%s\n",f(i))
60     printf("%s\n",f(i))
61 end
62 mclose(u)
63 halt('')
64 clc
65
66 halt(' ')
67  printf("$ cc signal2.c")
68  halt(' ')
69  printf("$ a.out ")
70  halt(' ')
71 printf("Press [Ctrl-z] first, then [Ctrl-c]")
72 halt("")
73 printf("# [Ctrl - \\] pressed                ")
74 sleep(2500)
75 printf("                    Signal Ignored  \n")
76 printf("# [Ctrl - z] pressed                \n")
77 sleep(2500)
78 printf("Cannot stop this program
     From tstp_handler\n")
79 printf("# [Ctrl - c] pressed                ")
80 sleep(2500)
81 printf("\nPress again                    From
     int_handler\n")
82 printf("# [Ctrl - c] pressed                \n")
```

```
sleep(2500)
printf("Quitting: Interrupted system call
   From int_handler")
halt(' ')
printf("\n\n\n$ exit          #To exit the current
   simulation terminal and return to Scilab console\
   n\n")
halt("........# (hit [ENTER] for result)")
//clc()

printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
    initial environment')
sleep(1000)
```

Scilab code Exa 24.16 Program 16

```
clear
flag=1
mode(-1)
clc
//Program for example 1 chapter 1
printf("Current date is %s \n \nWelcome to the
   Textbook Companionship Project 2013 \n",date())
printf("
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   n")
disp("Book Title                         :
         UNIX CONCEPTS AND APPLICATIONS ")
disp("Book Edition                       :

   4")
disp("Book Author                        :
```

```
    Sumitabha Das")
printf("
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    n")
disp("Code Author                           :

    Pranav Bhat T")
printf("
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    n")
disp("Chapter Number                        :

    24")
disp("Chapter Title                              :
                                          Systems programming
    II- Files")
printf("
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    n"
printf("Example 16     :         Show the method of
    using fork and exec to run a user-defined\n
    program and kill it in 5 seconds if not completed
     \n")
disp("
    ****************************************************************
    ")
disp("Answer     :    ")
disp("INSTRUCTIONS    : ")
halt(' ')
disp("1.These programs are part of systems
    programming PURELY in Unix and the commands have
    NO EQUIVALENT IN SCILAB")
halt(' ')
disp("2.However the .c files which are displayed
    here are also made into a seperate file.If you
    are a unix user then try compiling and running
    the programme with gcc or cc compiler")
halt(' ')
```

```
26 disp("3.The outputs displayed here are just MOCK
     OUTPUTS which are DISPLAYED IN THE TEXTBOOK")
27 halt(' ')
28 disp("4.The inconvenience is regretted.")
29 halt('.............Press [ENTER] to continue.....')
30 halt("")
31 clc
32 printf("\tUNIX SHELL SIMULATOR(DEMO VERSION WITH
     PRELOADED COMMANDS)\n\n\n")
33
34
35 i=0
36 i=i+1;f(i)='/* Program: killprocess.c -- Uses fork
     and exec to run a user-defined program and kills
     it if it doesnt complete in 5 seconds */'
37 i=i+1;f(i)='#include <stdio.h>'
38 i=i+1;f(i)='#include <sys/types.h>'
39 i=i+1;f(i)='#include <sys/wait.h>'
40 i=i+1;f(i)='#include <signal.h>'
41 i=i+1;f(i)=''
42 i=i+1;f(i)='pid_t pid;'
43 i=i+1;f(i)='int main(int argc, char **argv) {'
44 i=i+1;f(i)='    int i,status;'
45 i=i+1;f(i)='    void death_handler(int signo);    /*
     A common signal handler this time */'
46 i=i+1;f(i)='    '
47 i=i+1;f(i)='    signal(SIGCHLD, death_handler);   /*
     death_handler is invoked when a */'
48 i=i+1;f(i)='    signal(SIGALRM, death_handler);   /*
     child dies or an alarm is recieved */'
49 i=i+1;f(i)='    '
50 i=i+1;f(i)='    switch (pid = fork()) {'
51 i=i+1;f(i)='            case -1: printf('+ascii(34)+'
     Fork error\n'+ascii(34)+');'
52 i=i+1;f(i)='            case 0: execvp(argv[1],&argv
     [1]);   /* Execute command */'
53 i=i+1;f(i)='                    perror('+ascii(34)+'
     exec'+ascii(34)+');'
```

```
54 i=i+1;f(i)='                        break;'
55 i=i+1;f(i)='              default: alarm(5);       /*
     Will send SIGALRM after 5 seconds */'
56 i=i+1;f(i)='                        pause();       /*
     Will return when SIGCHILD signal is received */'
57 i=i+1;f(i)='                        printf('+ascii(34)+'
     Parent dies\n'+ascii(34)+');'
58 i=i+1;f(i)='                        }'
59 i=i+1;f(i)='              exit(0);'
60 i=i+1;f(i)='              }'
61 i=i+1;f(i)='    '
62 i=i+1;f(i)='    void death_handler(int signo) {
           /* This common handler pics up the */'
63 i=i+1;f(i)='             int status;
                            /* exit status for normal
     termination */'
64 i=i+1;f(i)='             signal(signo, death_handler);
       /* but sends the SIGTERM signal if */'
65 i=i+1;f(i)='             switch(signo) {
                           /* command doesnt complete in 5
      seconds */'
66 i=i+1;f(i)='                 case SIGCHLD: waitpid(-1,
    &status, 0); /* Same as wait(&status); */'
67 i=i+1;f(i)='                 printf('+ascii(34)+'Child
     dies; exit status: %d\n'+ascii(34)+',WEXITSTATUS(
     status));'
68 i=i+1;f(i)='                 break;'
69 i=i+1;f(i)='              case SIGALRM: if (kill(pid,
    SIGTERM) == 0)'
70 i=i+1;f(i)='                              fprintf(stderr,
     '+ascii(34)+'5 seconds over,child killed\n'+ascii
     (34)+');'
71 i=i+1;f(i)='                              }'
72 i=i+1;f(i)='}'
73 n=i
74
75
76 printf("\n\n$ cat killprocess.c      # to open the
```

```
    file emp.lst")
77  halt(' ')
78  u=mopen('killprocess.c','wt')
79  for i=1:n
80      mfprintf(u,"%s\n",f(i))
81      printf("%s\n",f(i))
82  end
83  mclose(u)
84  halt('')
85  clc
86
87  halt(' ')
88   printf("$ cc killprocess.c")
89   halt(' ')
90   printf("$ a.out  date")
91   halt(' ')
92   printf("Sat Jun 20  22:29:27  IST 2013\nChild dies:
        exit status: 0\nParent dies")
93  halt(' ')
94  printf("\n")
95  printf("$ a.out find /home -name a.out -print")
96  halt(' ')
97  printf("/home /sumit/personal/project8/a.out\n/home/
    sumit/personal/books_code/glass_ables/12/a.out\n/
    home/sumit/personal/books_code/stevens_c/ch08/a.
    out")
98  printf("  ..after 5 second time interval ...")
99  sleep(5000)
100 printf("\n5 seconds over, child killed\nParent dies\
    n")
101 halt(' ')
102 printf("\n\n\n$ exit        #To exit the current
    simulation terminal and return to Scilab console\
    n\n")
103 halt("........# (hit [ENTER] for result)")
104 //clc()
105
106 printf("\n\n\t\t\tBACK TO SCILAB CONSOLE...\nLoading
```

```
        initial environment')
107 sleep(1000)
```