DEPARTMENT

OF

ELECTRICAL ENGINEERING

Laboratory manual

for

PROGRAMMING LAB (EE233)

COLLEGE OF ENGINEERING
THIRUVANANTHAPURAM
695 016
JULY 2018

Preface

This is the first volume of the second edition of the Software laboratory manual released by the Department of Electrical Engineering, College of Engineering, Thiruvananthapuram. This is only a reference material, which will cater to the immediate needs of the students to meet the curriculum requirements of the B.Tech. Degree students. A manual covering all programs in this book is being prepared for the faculty for verification and reference. Every effort has been taken for the correctness of the subject dealt with, suggestions and remarks are welcome.

Chief Co-ordinator : **Dr. Sreejaya P.**
Head of the Department

Prepared and compiled by : **Dr. MAYADEVI N. & Prof. Shenil P. S**.
Assistant Professor

Department Vision and Mission

Vision

- Be a centre of excellence and higher learning in Electrical Engineering and allied areas.

Mission

- To impart quality education in Electrical Engineering and bring-up professionally competent engineers.
- To mould ethically sound and socially responsible Electrical Engineers with leadership qualities.
- To inculcate research attitude among students and encourage them to pursue higher studies

Syllabus

3rd Semester B.Tech (Electrical Engineering)

Year of Introduction 2016

EE 223 PROGRAMMING LAB 0-0-3

1. At least four simple programs using input output statements (example: area of rectangle, circle, etc)
2. At least four Simple programs using decision statements (Example: Even or odd, pass or fail)
3. Programs using control statements in C
4. Program to add n numbers
5. Programs to print patterns
6. Program to check whether a number is prime
7. Program to generate Fibonaacii series
8. Array manipulation (searching, insertion and sorting)
9. String handling Programs
10. Few programs using pointers
11. Functions – Pass by value, pass by reference
12. Handling Recursions.
13. File handling programs
14. Matrix manipulations – multiplication, inverse, determinants, transpose.
15. Solution of algebraic and transcendental equations: Bisection, Newton-Raphson method- comparison
16. Introductory programs using Python
17. Function calls in Python

Course Outcomes

KTU	Programme: B.Tech – Electrical and Electronics Engineering.	Semester: 3
Course Code: EE 233	Course Type: Lab	L-T-P: 0-0-3
Course Title: Programming Lab		
Credits: 1	**Assessment**	
	Internal Marks: - 100	University Exam Marks:-0 / Total Marks:100

Course Outcome	**At the end of the course, the student should be able to:**
CO1	Design, implement, test, debug, and document programs in C using basic input output and decision control statements
C02	Design, implement, test and debug programs in C using loop and switch statements
CO3	Build diversified solutions using array ,strings and matrices
CO4	Understand the concept of functions and build programs using functions and recursive functions
CO5	Design and develop C Programs with pointers and files.
CO6	Build basic programs using Python

Index

Coding Standard Used in the Software Lab

Program Header

```
// ************************************************
//      Program name                :              .c     *
//      Author                  :                        *
//      Date written                :                      *
//      Date complied               :                      *
//      Aim of the program      :                        *
//      Revision history            :                      *
//      Version         Author  Remarks                  *
//
//      0.1             XYZ     Initial draft            *
//      1.0             ABC     First release            *
//      1.1             XYZ     Modified on date         *
//                                                       *
//************************************************
//************************************************
```

Function Header

```
//************************************************
//      Function Name               :                      *
//      Purpose                     :                      *
//      Input Argument              :                      *
//
//      Return type                 :                      *
//************************************************
```

Expt 1: Conversion of temperature

Aim: Convert the temperature in degree Celsius to Fahrenheit

Algorithm

- Read temperature in degree Celsius into a variable C
- Calculate F using the formula F = 9/5 C + 32
- Display the result.

Expt 2: Find the largest of 3 numbers

Aim: To find the largest of three numbers

Algorithm

- Read 3 numbers A,B,C
- Assign Large = A
- Check if B > Large , if Yes Large = B
- Check if C > Large , if Yes Large = C
- Display the value of large as the largest number

Expt 3: Square a number with LSB 5

Aim: To calculate the square of the number entered by the user, if the least significant digit of that number is 5

Algorithm

- Read the number
- With Modular division operation, last digit is separated to confirm whether its 5 or not
- If it is '5', then square the number, else display LSB not 5 and exit
- Display the result

Expt. 4: Swap the values of two variables

Aim: To swap the values of two variables without using a third variable

Algorithm

- Read and assign to variables
- Sum of variables stored to the 1st variable
- Difference of the current 1st variable and 2nd variable is stored as the new 2nd variable
- The difference of the current variables are stored as the 1st variable
- Display the result

Expt 5: Convert an integer to a character

Aim: To convert an integer to a character

Algorithm

- Declare a variable as type integer
- Read its value through the keyboard
- The entered number and ASCII value are checked
- If there is a match, display the ASCII value

Expt. 6: Count specific numbers

Aim: To count numbers between 1 and 100 which are not divisible by 2, 3 and 5

Algorithm

- Numbers from 1 to 100 are checked in a loop
- Each time perform mod operation with 2,3 and 5
- If the remainder is not zero, increment the counter
- End loop
- Display the counter, as the result

Expt 7: Perform arithmetic functions

Aim: To perform various arithmetic functions such as Addition, Subtraction, Multiplication, Division, Remainder Calculation etc.

Algorithm

- Display a menu with 5 options and request the user to enter the choice
- Go to suitable sections based on the entered choice
- Prompt the user to enter two numbers
- Perform the relevant operation
- Display the result
- Warn the user with "Invalid Choice", if the entered choice is wrong

Expt 8: Check a number for prime or not

Aim: To check whether the given number is prime or not

Algorithm

- Enter the number n
- Perform the 'mod' operation from 2 to (n-1).
- If remainder is zero, 'n' is not a prime number.
- If remainder is non-zero, then 'n' is a prime number

Expt 9: Evaluate sine and cosine series

Aim: To evaluate sine and cosine series

Algorithm

(a) Sine Series

- Input the angle in degrees and number of terms as X and N respectively
- Convert angle to radians and assign it to X
- Assign the 1st term as angle in radians
- Initialize the sum variable as 1st term
- Vary a counter from 2 to N-1 with an increment two (Set the counter as i)
- Set the increment value as Incr
- Update Incr with – Incr *X^2/ (i * (i + 1)
- Add sum with Incr until i = N - 1
- End loop
- Display sum as the result

(b) Cosine Series

- Input the angle in degrees and the number of terms as X and N respectively
- Convert angle to radians
- Initialize the sum variable as 1
- Vary a counter from 2 to N with an increment 2
- Set the counter as i
- Set the increment values as Incr

- Update Incr = Incr *X^2/ (i * (i - 1)
- Add sum with Incr until i = N
- End loop
- Display sum as the result

Expt 10: Solve a Quadratic Equation

Aim: To solve a quadratic equation

Algorithm

- Read in the values of the coefficients as A, B and C
- Evaluate D = B^2– 4 * A * C
- Check if D>0

  If Yes , the 1st root = R1 = -B + $\sqrt{D}$ /2A

  2nd root = R2 = -B + $\sqrt{D}$ /2A
- If D=0

  R = -B /2A,

  Display roots
- If D < 0

  Display the roots as imaginary

Expt. 11: Sort an array in ascending order

Aim: To sort an array of N integers in ascending order

Algorithm

- Store N integers in an array
- First element taken, compared with all other elements of the same array, the process is continued till the last element, in a loop
- Interchange the elements if required
- End loop
- Display the final sorted array

Expt 12: Evaluate the length of a string

Aim: To find the length of a given string

Algorithm

- Read in the string through the keyboard
- Count the number of characters in the string using a loop
- Display the value of counter as the length

Expt 13: Copy contents of a string to another

Aim: To copy the contents of a string to another without using a library function

Algorithm

- Declare two arrays of same length
- Use a loop and copy each character of the source string to the destination string
- Display both the strings

Expt 14: Compare two strings

Aim: To compare two strings without any standard function and if the strings are not identical, display the position where the characters are different

Algorithm

- Read in the two strings
- Using a loop, check the corresponding characters of both the strings
- If identical, loop is continued
- Otherwise, a counter variable 'diff' is incremented
- Display the locations where the strings are non identical
- Display the no: of places, where two strings are different

Expt 15: Concatenate two strings

Aim: To concatenate two strings without a standard function

Algorithm

- Read in the two strings
- Declare a 3rd string to store the concatenated string
- Use a loop, store the 1st string to the third
- Store the 2nd string to the 3rd, starting from the current position
- Display the third string

Expt 16: Reverse a given string

Aim: To reverse a given string

Algorithm

- Read in a string, 'str'
- Declare a 2nd string, 'rev_str'
- Using a loop, position a variable 'i' to the last position of original string (not the null character
- Assign j=0
- Copy the contents of the given string 'str' onto the reverse string 'rev_str', using the relation rev_str [j] = str [i]
- Increment j and decrement i till i>=0
- Display rev_str as the result

Expt 17: Check a palindrome string

Aim: To check whether the given string is a palindrome or not

(Hint: Palindrome word reads same from left to right and right to left)

Algorithm

- Read in a string
- String length is found and assigned to a variable
- Use a condition inside a loop to check for the 1st and last character equality
- 2 Separate variables are used to point to the first and last characters
- One variable incremented and the other decremented
- If found unequal, exit – display not palindrome
- If found equal, display a palindrome

Expt 18: Generate Fibonacci series

Aim: To generate a Fibonacci series

Algorithm

- Read n, the number of terms to be generated.
- Term 1 = 0, term2 = 1.
- Display term1, term2.
- Vary the counter in a loop from 0 to (n – 1) incremented by 1.

- Term = term1 + term2.
- Display term.
- Term 1 =term2.
- Term 2 = term -->
- End loop

Expt 19: Factorial of a given number

Aim: To find the factorial of a given number using recursion

Algorithm

- Read the value of 'n', whose factorial is to be found
- Function call F=factorial (n);
- Inside the function factorial (x) define a local variable x
- If (X==1) return 1
- Else fact = x* fact (x-1)
- Return fact
- Display fact

Expt 20: Subtraction of matrices

Aim: To read two matrices of order 2 x 2 and subtract one from the other using functions

Algorithm

- Pass the 2 dimensional arrays as argument to functions
- Size of row is optional, but the column size is must
- Read the first array using 'input' function (use arrays)
- Input the 2nd array using the same function (use arrays)
- Subtraction is done by passing the 2 matrices / arrays to the 'sub' function
- Element by Element Subtraction is done inside the 'sub' function
- Display the resultant matrix

Expt 21: Transpose a matrix

Aim: To find the transpose of an 'n x n' matrix

Algorithm

- Read in the size of the matrix
- Read in he matrix elements (using arrays)
- Interchange Rows and Columns, using two loops

  (Hint: The rows elements become column elements and vice versa)
- Display the row elements as columns elements
- Display the column elements as row elements

Expt 22: Multiplication of two matrices

Aim: To multiply two matrices

Algorithm

- Read in the order of matrix A as M and N
- Read the order of matrix B as N1 and L
- If N1 = ! N, display multiplication is not possible
- Display error message and prompt to enter the new size
- Read in the elements of A matrix into an array
- Read in the elements of B matrix into an array
- Find C = A * B (Use proper loops)
- Display the matrix C

Expt 23: Inverse of a matrix

Aim: To find the inverse of a matrix

Algorithm

- Read in the order of the matrix (n x n) say 'A'
- Read in the matrix elements
- Formula used is $A A^{-1} = I$ (identity Matrix)
- Let B denote the matrix in RHS
- Assign B = I

- Do the required transformation for A and continue the same for B also, so that finally A is reduced to identity matrix
- Display matrix B, which will be the inverse of A

Expt 24: Example to demonstrate working of pointers

Aim: To demonstrate working of pointers

Algorithm

- Define the pointer
- Display the address and value of the defined variable
- Store address of the variable to pointer
- Display the address and value of the pointer
- Assign a new value to the memory that addressed by the pointer (the variable)
- Display the Display change in the value of the variable

Expt 25: Swap two integers using functions

Aim: To swap two integers using functions by Pass by value and Pass by reference

Algorithm

- Function named 'swap' which accepts two arguments declared
- (Hint :swap (x ,y))
- 'swap' is called in the main program in 2 ways

  (a) The actual value of x and y are copied to the function where swapping is done

  (b) The swapping is reflected in the actual parameters by passing reference. The original variable is passed and not a copy

  .

Expt 26: Length of a character string using pointers.

Aim: Program using pointers to determine the length of a character string

Algorithm

- Define a pointer
- Initialize a character string
- While loop defined to find out the end of the character string
- Each element in the string along with their address displayed with the help of while loop
- Length of the string found out from the difference between the pointers
- Display the result

Expt 27: Sum of all elements stored in an array using pointers

Aim: Program using pointers to compute the sum of all elements stored in an array

Algorithm

- Define a pointer
- Read array elements
- Perform addition of all elements stored in an array, using pointers
- Display the result

Expt 28: Operations with a file

Aim: Write a program which create a file for students (name, Roll No, marks) and then perform the following operation

(1) Store the name, Roll No and marks of 5 students onto file

(2) Print the contents of the file

(3) Add one student information into file

(4) Display number of students in the file

(5) Modify any student information

Algorithm

- Declare the Struct 'students' with all information of student and with functions 'getdata' and 'printdata'
- Open file, call the 'getdata' function and store in the data
- Write to the file the data using a function – 'write-to-file()'

- Turn off the EOF flag
- Add one more student data use function 'add-data-to-file ()'
- Display the appended file from beginning
- Using 'seek' function, modify the student file
- Display the updated file

Expt 29: Solution of Differential Equation (Euler's Method)

Aim: To solve a differential equation using Euler's method

Algorithm

- Define the differential Equation df (x , y)
- Input values of x0, y0, h, x

  (Hint: x0 → initial value of x

  y0 →initial value of y

  h → increment

  x → final value of x for which y has to be found)
- Assign x1 = x0
- Assign y1= y0
- Is x1 > x0 If Yes-stop
- Else, Set y1 = y1 + h * df (x1 , y1)
- Set x1 = x1 +h
- Print x1, y1
- Continue the loop till x1 >x is true

Expt 30: Solution of Differential Equation (Runge-Kutta method)

Aim: To solve a differential equation Runge-Kutta method

Algorithm

- Define the function f (x, y)
- Input values of x_0, y_0, h, x_n
- Is $x > x_n$ if Yes Stop.
- Else, Set k1 = h * f (x, y)
- Set k2 = h * f (x+h/2 , y + k1 /2)
- Set k3 = h * f (x+h/2 , y + k2 /2)
- Set k4= h * f (x+h, y + k 3)
- K = (k1 + (k2 +k3) *2 + k4) /6
- Set x= y + h
- Set y = y + k
- Print x and y
- Continue the loop till $x = x_n$

Expt 31: Solve by numerical integration (Trapezoidal rule)

Aim: To solve numerical integration by trapezoidal rule

Algorithm

- Y (x) is the function to be integrated
- Get values of x0 , xn and n

  x_0 → initial value

  X_n → final value

  n → no. of intervals
- Assign h = (x_n– x_0) /n
- Assign s = y (x_0) + y (x_n)
- Loop for i =1 to n-1
- Assign s = s +2 * y (x_0+ i* h)
- Endloop i
- Print the solution as s * h /2

Expt 32: Solve by numerical integration (Simpson's rule)

Aim: To solve a numerical integration by Simpson's rule

Algorithm

Y(x) is the function to be integrated

- Get values of x_0, x_n, and n
- Assign $h = (x_n - x_0) / n$
- Assign $s = y_0 + y_n + 4y_1$
- Loop for i = 3 to n-1 in steps of 2
- Assign $s = s + 4*y_i + 2 * y_{(i-1)}$
- End loop i
- Print the solution as s * h / 3

Expt 33: Find root of an equation (Newton-Raphson method)

Aim: To find the root of an equation using Newton-Raphson method

Algorithm

- Define the equation F (X) = 0 whose root is to be found
- Define d f (x), the derivative of f (x) w.r.t x
- Input the values of x_0, aerr, maxitr
- (Hint aerr = allowed error)
- Loop for itr = 1 to maxitr

- Assign h = f (x_0) / df (x_0)
- x_1= x_0 -
- Print itr and x1
- If (h< aerr) yes, print solution and stop
- Else
- Assign x_0= x_1
- End loop itr
- Print solution does not converge and stop

Expt 34: Find root of an equation (Bisection Method)

Aim: To find the root of an equation using Bisection Method

Algorithm

- Define the function
- Read x0,x1 and e
- y0 = f(x0)
- y1=f(x1)
- if sign(y0)=sign(y1) then stop
- Else while| (x1-x0)/x1 | > e do steps 7 to 9
- x2 = (x0+x1)/2
- y2 =f(x2)
- i=i+1
- if sign(y0)=sign(y2) then x0 =x2 else x1=x2
- write x2,y2

Expt 35: Ohms Law Calculator

Aim: To write a python program to calculate voltage

Algorithm

- Give the user directions
- Read current in amps
- Read the resistance in ohms
- Compute the voltage from V=I*R
- Print out the resulting voltage in volts

Expt 36: Factorial Program

Aim: To find factorial of a given number.

Algorithm

- Read input number
- Check whether the input number is negative and show appropriate message
- If the number entered is 0, print factorial as 1
- When the entered number is positive integer and greater than 1, initialize a variable factorial as 1 then run a for loop from 1 to the number+1
- Calculate the factorial as factorial * i
- Print the result

Expt 37: Sum of Items In a List

Aim: To find the sum of all the elements in a list.

Algorithm

- Read input number asking for length of the list
- Initialise an empty list lst = [].
- Read each number using a for loop.
- In the for loop append each number to the list.
- Use predefined function sum() to find the sum of all the elements in a list.
- Print the result

Expt 37: Search for an Items In a List

Aim: Using linear search to find the index/position of the target K in list L

Algorithm

- Initialize a boolean variable found and set it to False initially
- Start for loop with i ranging from 0 to the length of the list
- Check if key element is equal to L[i]
- if equals
- set found boolean variable to True
- print the position of the element found
- break for loop
- If the boolean variable found is equal to False after for loop, print that the element is not found in the list

Expt 37: Print *Fibonacci sequence*

Aim: To print Fibonacci sequence for a given input.

Algorithm

- Read input number for which the Fibonacci sequence is to be found
- Uisng for loop ranging from 0 to the input number and call fibonacci() function which returns the output of the Fibonacci sequence.
- Function fibonacci()
- check if the input number is 1, if 1 then return 1
- check if the input number is 0, if 0 then return 0
- if input number n is > 1, again call fibonacci function with it next 2 numbers (n-1, n-2)
- Print the result from Fibonacci function, which prints the required Fibonacci sequence

Standard References

1. E. Balaguruswamy, Programming in ANSI C, Tata McGraw Hill, New Delhi
2. Kernighan, Brian W., and Dennis M. Ritchie. The C programming language. Vol. 2. Englewood Cliffs: prentice-Hall, 1988.
3. Introduction to computation and programming using Python, John V. Guttag, PHI Learning, New Delhi
4. Downey, Allen, Jeffrey Elkner, and Chris Meyers. How to think like a computer scientist: learning with python. John Wiley 2015.
5. Lambert, Kenneth. Fundamentals of Python: first programs. Cengage Learning, 2011.
6. Yaswant Kanitkar, "Let us C", BPB Publications.
7. Ashok N. Kamthane, "Programming with *ANSI* and TURBO C", Pearson Education
8. Schaums Outline series to C Programming
9. Stephen G Kochan , "Programming in C", CBS Publishing Co.
10. Brian W Kernighan & Dennis M Ritchie, "The C Programming language" Pretice Hall - India-1986
11. W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, "Numerical Recipes in C"