VISHNU
UNIVERSAL LEARNING

COMPUTER PROGRAMMING THROUGH C LAB MANUAL

Name	
Roll No.	
Branch	
Section	

INDEX

OBJECTIVES OF THE LAB

- To make the student learn a programming language.
- To teach the student to write programs in C solve the problems
- Student learn the concepts like looping ,functions ,pointers, file .concepts

Primary goal of this course is to make acquaint the students to know the programming language and also to know how 'C' can be used to write serious program on IBM. Here the main importance is given to the knowledge and concepts that are needed to exploit the capabilities of the microcomputer through C.

More over programming is a 'ART' purely depends on the logic of the problem solving . As it is said that problem solving has many methods, but the requirement is the efficient way of solving a problem.

And more over the language C contains the control structures necessary to make programmers readable and also allows basic concepts like looping ,functions ,pointers, file. Concepts is to be implemented in variety of ways.
In this course we mainly concentrate on programming concepts of C and its implementation.

REQUIREMENTS

Hardware Requirements:

Processsor	Pentium I
RAM:	64MB
Hard Disk	100 MB

Software Requirements:

Language:	ANSI C Compiler with Supporting Editors

Hard ware Requirement to run C language

Processor	PIV(166 MHZ)
RAM	64 MB
HDD	100 MB
Monitor	14''Color
Keyboard	Normal

LAB SYLLABUS PROGRAMS

Week 1:

a. Design flow charts for logical problems.
b. Design algorithm for simple problems.

Week 2:

a. Design Pseudocode steps for simple problems.
b. Write a simple program based on operators (pre, post increment , bitwise and , or , etc.).
c. Write a simple program based on conversions (from int to float & float to int)

Week 3:

a. Write a program for find the max and min from the three numbers.
b. Write the program for the simple, compound interest.
c. Write program for students marks grading.

Week 4:

a. The total distance travelled by vehicle in 't' seconds is given by distance = ut+1/2at2 where 'u' and 'a' are the initial velocity (m/sec.) and acceleration (m/sec2). Write C program to find the distance travelled at regular intervals of time given the values of 'u' and 'a'. The program should provide the flexibility to the user to select his own time intervals and repeat the calculations for different values of 'u' and 'a'.
b. Write a C program, which takes two integer operands and one operator from the user, performs the operation and then prints the result. (Consider the operators +,-,*, /, % and use Switch Statement)

Week 5:

a. Write a C program to find the sum of individual digits of a positive integer and test given number is palindrome.
b. A Fibonacci sequence is defined as follows: the first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Write a C program to generate the first n terms of the sequence.
c. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.

Week 6:

a. Write a C program to calculate the following
    i. sum: sum=1-x2/2! +x4/4!-x6/6!+x8/8!-x10/10!
    ii. sum=x-x^3/3!+x^5/5!........................,
    iii. sum=1+x/1!+x^2/2!+x^3/3!.............,
b. Write a C program to find the roots of a Quadratic equation.

Week 7:

a. Write C programs that use both recursive and non-recursive functions
    i. To find the factorial of a given integer.
    ii. To find the GCD (greatest common divisor) of two given integers.
b. Write a program for implementing of Storage classes: (Auto, static, extern, register)

Week 8:

a. Write a C program to find the minimum and maximum integer in a list of integers.
b. Write a C program that uses functions to perform the following:
   i. Addition of Two Matrices
   ii. Multiplication of Two Matrices
   iii. Transpose of a matrix

Week 9:

a. Write a C program that uses functions to perform the following operations:
   i. To insert a sub-string in to a given main string from a given position.
   ii. To delete n Characters from a given position in a given string.
b. Write a C program to determine if the given string is a palindrome or not

Week 10:

a. Write a C program that displays the position or index in the string S where the string T begins, or – 1 if S doesn't contain T.
b. Write a C program to count the lines, words and characters in a given text.

Week 11:

a. Write a C program to generate Pascal's triangle.
b. Write a C program to construct a pyramid of numbers.

1	*	1	1	1	*
1 2	* *	2 3	2 2	0 1	* *
1 2 3	* * *	4 5 6	3 3 3	0 1 0	* * *
			4 4 4 4	1 0 1 0	* *
					*

Week 12:

a. Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression:
   1+x+x2+x3+………….+xn
   For example: if n is 3 and x is 5, then the program computes 1+5+25+125. Print x, n, the sum
   Perform error checking. For example, the formula does not make sense for negative exponents – if n is less than 0. Have your program print an error message if n<0, then go back and read in the next pair of numbers of without computing the sum. Are any values of x also illegal? If so, test for them too.

b. 2's complement of a number is obtained by scanning it from right to left and complementing all the bits after the first appearance of a 1. Thus 2's complement of 11100 is 00100. Write a C program to find the 2's complement of a binary number

Week 13:

c. Write a functions to compute mean , variance , SD, sorting of n elements in single dimension array.
d. Write a C program to convert a Roman numeral to its decimal equivalent.

Week 14:

a. Write a program for reading elements using pointer into array and display the values using array.

b. Write a program for display values reverse order from array using pointer.
c. Write a program through pointer variable to sum of n elements from array .

Week 15:

a. Write a C program which copies one file to another.
b. Write a C program to reverse the first n characters in a file. (Note: The file name and n are specified on the command line.)

Week 16:

a. Write a C program to display the contents of a file.
b. Write a C program to merge two files into a third file (i.e., the contents of the first file followed by those of the second are put in the third file)

Outcomes : After completion of the course, the students would be able to:

- Understand the basic terminology used in computer programming
- Write, compile and debug programs in C language
- Design programs involving decision structures, loops and functions

INTRODUCTION ABOUT LAB

Program Development Steps

To develop the program in high level language and translate into machine level language we are having following steps.

1) Writing and editing the program
2) Compiling the program
3) Linking the program with the required library modules
4) Executing the program

Algorithm:-

It is method of representing the step by step process for solving a problem.
Each step is called an instruction.
Characteristics of algorithm

- **Finiteness :-** It terminates with finite no of steps
- **Definiteness:-** each step of algorithm is exactly defined
- **Effectiveness:-**All the operations used in the algorithm can be performed exactly in a fixed duration of time
- **Input:-**An algorithm must have an input before the execution of program begins
- **Output:-** An algorithm has one or more outputs after the execution

Example of algorithm to find sum of two numbers:

```
Step1:  Begin
Step2:  read a,b
Step3:  add a and b and store in variable c
Step4:  display c
Step5:  stop
```

History of C

The *milestones* in C's development as a language are listed below:

- UNIX developed c. 1969 -- DEC PDP-7 Assembly Language
- BCPL -- a user friendly OS providing powerful development tools developed from BCPL. Assembler tedious long and error prone.
- A new language ``B'' a second attempt. c. 1970.
- A totally new language ``C'' a successor to ``B''. c. 1971
- By 1973 UNIX OS almost totally written in ``C''.

ABOUT C LANGUAGE :

C is a programming language developed at AT&T's BELL Laboratory of USA in 1972. Dennis Ritchie designed it. Because of its reliability. C is very popular. C stands between high-level language (HLL) and Low Level Language (LLL). C's compactness and coherence is mainly due to the fact that it is a one-man language.

C is highly portable & it is well suited for structured programming. C program consists of collection of functions.

STRUCTURE OF 'C' PROGRAM :

Each instruction in a C program is written as a separate statement. The program starts with a main function followed by the opening braces which indicates the start of the function then the variable and constant declarations followed by the statements, which includes input and output statements.

C program may contain one or more sections as shown below.

```
DOCUMENTATION SECTION

LINK SECTION

DEFINITION SECTION

GLOBAL DECLARATION SECTION

Main() Function section

{

        Declaration part

        Executable part

}

SUBPROGRAM SECTION

        User defined functions
```

Keywords

C89 has 32 keywords (reserved words with special meaning): auto, break, case, char, const, continue, default, do, double, else, enum, extern, float, for, goto, if, int, long, register, return, short, signed, sizeof, static, struct, switch, typedef, union, unsigned, void, volatile, and while.

C99 adds five more keywords: inline, restrict, _Bool, _Complex, and _Imaginary.

Operators

C supports a rich set of operators, which are symbols used within an expression to specify the manipulations to be performed while evaluating that expression. C has operators for:

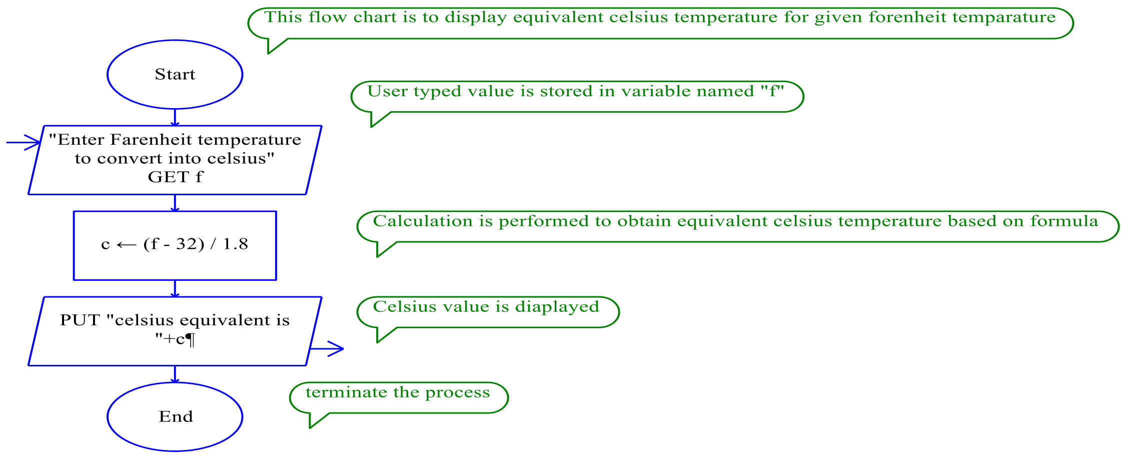
- arithmetic: +, -, *, /, %
- assignment: =
- augmented assignment: +=, -=, *=, /=, %=, &=, |=, ^=, <<=, >>=
- bitwise logic: ~, &, |, ^
- bitwise shifts: <<, >>

- boolean logic: !, &&, ||
- conditional evaluation: ? :
- equality testing: ==, !=
- calling functions: ( )
- increment and decrement: ++ and --
- member selection: ., ->
- object size: sizeof
- order relations: <, <=, >, >=
- reference and dereference: &, *, [ ]
- sequencing: ,
- subexpression grouping: ( )
- type conversion: (*typename*)

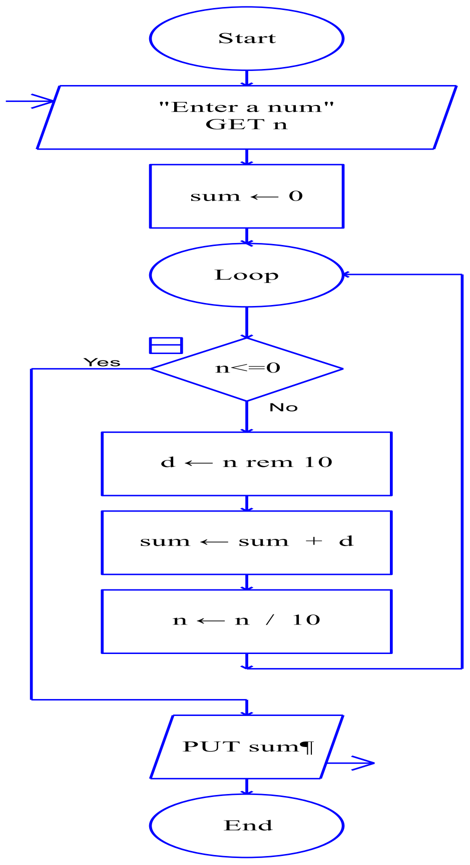
SOLUTIONS FOR PROGRAMS

Week 1:

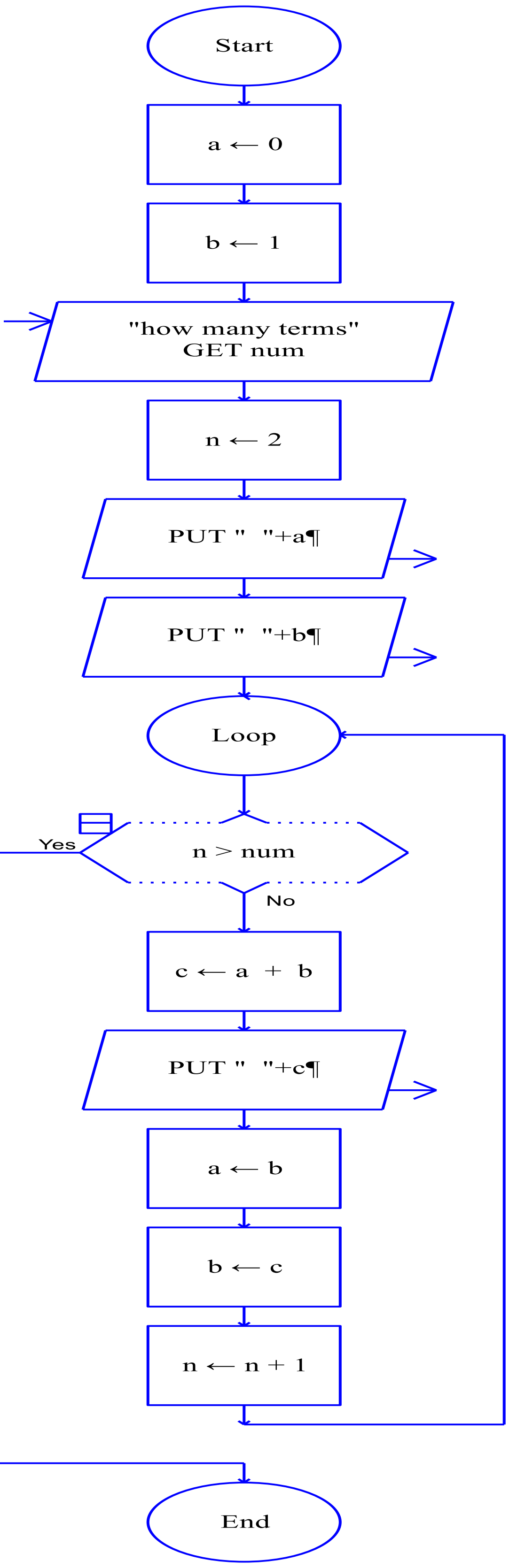
A. Design flow charts for logical problems.
Example :-


This flow chart is to display equivalent celsius temperature for given forenheit temparature
Start
User typed value is stored in variable named "f"
"Enter Farenheit temperature to convert into celsius" GET f
c ← (f - 32) / 1.8
Calculation is performed to obtain equivalent celsius temperature based on formula
PUT "celsius equivalent is "+c¶
Celsius value is diaplayed
End
terminate the process

Draw Flow Chart to find sum of consisting digits of a no.



Flowchart to print n terms of Fibonacci sequence


Start
a ← 0
b ← 1
"how many terms"
GET num
n ← 2
PUT " "+a¶
PUT " "+b¶
Loop
n > num
Yes
No
c ← a + b
PUT " "+c¶
a ← b
b ← c
n ← n + 1
End

B. Design algorithm for simple problems.

Example:

```
Step 1: Start
Step 2: read forenheit temperature
Step 3:calculate equivalent centigrade temperature using formula
Step 4:display centigrade temperature
Step 5:stop
```

Week 2:

a. Design Pseudo code steps for simple problems.
   - Determine INPUT
   - Determine OUTPUT
   - Determine processing
   - Statements are written in simple English.
   - Each instruction is written on a separate line.
   - Keywords and indentation are used to signify particular control structures.
   - Each set of instructions is written from top to bottom with only 1 entry and 1 exit.
   - Groups of statements may be formed into modules and that group can be given a name

   For example :Swap Two Variables
   - Take two variables a and b
   - Where a = 4 and b = 5,
   - In swap two numbers we will use a temp variable
   - After swapping a value is 5, b value is 4

b. Write a simple program based on operators (pre, post increment , bitwise and , or , etc.).

   Using operators and operands check the results.

c. Write a simple program based on conversions (from int to float & float to int)

   Using various types of data variable check the promotion of the variables and type casting method.
   1. char and short operands are converted to int
   2. Lower data types are converted to the higher data types and result is of higher type.
   3. The conversions between unsigned and signed types may not yield intuitive results.
   4. Example

      float f; double d; long l;

      int i; short s;

d + f	f will be converted to double
i / s	s will be converted to int
l / i	i is converted to long; long result

- The general form of a type casting operator is
  (type-name) expression
- It is generally a good practice to use explicit casts than to rely on automatic type conversions.
- Example
  - C = (float)9 / 5 * (f – 32)
- float to int conversion causes truncation of fractional part
- double to float conversion causes rounding of digits
- long int to int causes dropping of the higher order bits

Week 3:

a. Write a program for find the max and min from the three numbers.

```
Step 1: Start
Step 2: read a, b, c
Step 3: check a > b and b > c if yes goto step 4 else goto step 5
Step 4: display a is max and c is min goto step 9
Step 5: check b > c and c > a if yes goto step 6 else step 7
Step 6: display b is max and a is min goto step 9
Step 7: check c > a and a > b if yes goto step 8 else step 9
Step 8: display b is max and a is min goto step 9
Step 9: stop
```

b. Write the program for the simple, compound interest.

```
Step 1: Start
Step 2: read principal,time,rate
Step 3: calculate simple interest using formula store result in si
Step 4: calculate compound interest using formula store result in ci
Step 5: display simple interest value
Step 6: display compund interest value
Step 7: stop
```

c. Write program for students marks grading.

```
Step 1: Start
Step 2: read subject1, subject2, subject3, subject4
Step 3: calculate sum of the all subjects into sum and find average
Step 4: check average more than 75 then print distinction
```

```
        other wise average more than 60 then print first class
        other wise print pass
    Step 5: stop
```

Week 4:

a. The total distance travelled by vehicle in ‘t’ seconds is given by distance = ut+1/2at2 where ‘u’ and ‘a’ are the initial velocity (m/sec.) and acceleration (m/sec2). Write C program to find the distance travelled at regular intervals of time given the values of ‘u’ and ‘a’. The program should provide the flexibility to the user to select his own time intervals and repeat the calculations for different values of ‘u’ and ‘a’.

ALGORITHM:

```
Step 1: Start

Step 2: ch= `Y’

Step 3: if ch  is not equal to ‘Y’ go to 10

Step 4: Input the values for u, a, t (time interval) and   T (total time)

Step 5: repeat steps 6, 7 and 8  T times

Step 6: s = ((u*t)+(0.5*a*t*t))

Step 7: t = t +i

Step 8: Print the  value of  s   go to step  5

Step 9: Read a character for ch  and  go to step  3

Step 10: Stop
```

Test Data:

INPUT

u =3 , a =2 , T= 4, t= 1

OUTPUT

S= 28.0000   ch =   ‘N ‘

b. Write a C program, which takes two integer operands and one operator from the user, performs the operation and then prints the result. (Consider the operators +,-,*, /, % and use Switch Statement)

ALGORITHM:

```
Step 1: Start
Step 2: Read two values a, b.
Step 3: Input an operator to be performed.
Step 4: operator matches with
    Step a) `+' : c = a+ b, go to 5

    Step b) `-' : c = a-b,  go to 5

    Step c) `*' : c= a*b, go to 5

    Step d) `/' : c = a/b , go to 5

    Step e) `%': c= a % b go to 5

Step 5:   print c
Step 6:   Stop
```

Test Data:

INPUT

a=2, b=3 Operator = ` +'

OUTPUT

c =5

Week 5:

a. Write a C program to find the sum of individual digits of a positive integer and test given number is palindrome.

ALGORITHM:

```
Step 1: Start

Step 2: Input the value of n

Step 3: if n is equals to zero goto 9

Step 4: mod(n,10) store result in rem

Step 5: add rem value to sum

Step 6: divide n by 10 then store result in n

Step 7: go to step 3

Step 8: print the value of sum

Step 9: stop
```

Test Data:

INPUT

n=213

OUTPUT

```
6
```

b. A Fibonacci sequence is defined as follows: the first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the preceding two terms in the sequence. Write a C program to generate the first n terms of the sequence.

ALGORITHM:

```
Step 1: Start
Step 2: Input a value for n.
Step 3: Initialize   the variables    a=0, b=1,c=0, I=3
Step 4: Print the values of a and b
Step 5: c = a + b
Step 6: Print the value of c
Step 7: Assign a= b and b=c
Step 8: add unit value to I
Step 9: if I less than or equal to n, go to step 5
Step 10: stop
```

Test Data:

INPUT

n = 8

OUTPUT

```
0  1  1  2  3  5  8  13
```

c. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.

ALGORITHM:

```
Step 1: Start
Step 2: Input a value for n
Step 3: repeat steps 4 to 7 for i= 1 to n
Step 4: Initialize flag =0
Step 5: for j = 2 to i-1 repeat step 6
Step 6:  if remainder of i / j is equal to zero
Step 7:  flag =flag + 1
Step 8: if flag is equal to zero
Step 9: Print the value of i
Step 10. Stop
```

Test Data:

INPUT

n =10

OUTPUT

1 2 3 5 7

Week 6:

a. Write a C program to calculate the following

   i. sum: sum=1-x2/2! +x4/4!-x6/6!+x8/8!-x10/10!

ALGORITHM:

```
Step 1 :  Start
Step 2 : Input a value for x
Step 3  :  sum=1 , I= 2 , t = 1
Step 4  :  If   i>10 goto 8
Step 5  : t = (-1)*( (x ^2)/((i-1)*i))
Step 6:  increment the value of  I by 2
Step 7 :   sum = sum +t  , goto 4
Step 8 : Print the value of sum
Step 9:  stop
```

Test Data:

INPUT

x = 1

OUTPUT

Sum = 0.5

   ii. sum=x-x^3/3!+x^5/5!........................,

```
Step 1: start
Step 2:enter the number of terms n, value x
Step 3: term value assign as x,sum as a  x
Step 4: index  i  value 1
Step 5: start loop until i<=n
Step 6: term value is -(term * x * x/2i(2i+1))
Step 7: sum value is sum+term
Step 8: i value is incremented by one
Step 9:  stop loop
Step 10: stop
```

iii. sum=1+x/1!+x^2/2!+x^3/3!............,

```
Step 1:: start
Step 2:  enter the number of terms n, value x
Step 3: term value assign as 1,sum as a  1
Step 4: index  i  value 1
Step 5: start loop until i<=n
Step 6: term value is (term * x /i)
Step 7: sum value is sum+term
Step 8: i value is incremented by one
Step 9: stop loop
Step 10: stop
```

b. Write a C program to find the roots of a Quadratic equation.

ALGORITHM:

```
Step 1:  Start
Step 2:  Input the  values for a, b, c
Step 3:  d=b*b –4*a*c
Step  4: if d greater than zero
Step 5:  a)print roots are read and distinct
Step 6: r1=  -b+ √ d  / (2*a)
Step  7: r2= -b-√ d  / (2*a)

Step  8: Print r1 and r2 , go to 7

Step  9: if d is equal to zero

Step 10:  Print roots are real and equal

Step 11: r1 = - b/2*a

Step 12: Print  r1,    go to 7

Step 13:  else

Step 14:  Print roots are imaginary

Step 15: stop
```

Test Data:

INPUT

a=2, b=3, c=4

OUTPUT

roots are imaginary

Week 7:

a. Write C programs that use both recursive and non-recursive functions
   To find the factorial of a given integer.

ALGORITHM: (NON-RECURSIVE)

```
Step 1: Start

Step 2: Input the value for n

Step 3: Calling the function factorial

Step 4:  P= factorial  (n)

Step 5:  Print    P

Step 6:  Stop
```

Procedure factorial (x)

```
Step 1:  Start

Step 2:  fact =1

Step 3:  if i > x go to 5

Step 4: fact = fact *i
Step 5: i  =i+1 , go to step 2
Step 6:  Return factorial. (procudure termination)
```

Test Data:

INPUT

n=5

OUTPUT

120

ALGORITHM: (USING RECURSION)

```
Step 1: Start
Step 2: Input the value for n
Step 3: P= factorial (n)
Step 4: print p
Step 5: Stop
```

Procedure factorial(x)

```
Step 1:. Begin
Step 2: if x is equal to 1 then return 1
Step 3: if not
Step 4: f = x  * factorial (x-1)
Step 5: return f
```

Test Data:

INPUT

n=5

OUTPUT

120

b. **To find G.C.D of two numbers**

ALGORITHM: **(USING NON-RECURSION**)

```
Step 1: Start
Step 2: Input the values for a and b
Step 3: x= gcd (a ,b)
Step 4: print the value of x
Step 5: Stop
```

Procedure gcd (i , j)

```
Step 1: Begin
Step 2: temp  = remainder of   i / j
Step 3: assign  i=j , j= temp
Step 4: if temp not equal to zero go to step 2
Step 5: return  j
```

Test Data:

INPUT
36 , 8
OUTPUT
4

ALGORITHM : (USING RECURSION)

```
Step 1: Start
Step 2: Input the values for a and b
Step 3: x= gcd ( a ,b)
Step 4: Print the value of x
Step 5: Stop
```

Procedure gcd (i, j)

```
Step 1: Begin
Step 2: if  j is equal to zero , return i
Step 3: if not
Step 4: r = remainder of  i / j
Step 5: b)  return   gcd (j, r)
Step 6: end
```

Test Data:

INPUT
36 , 8
OUTPUT
4

b. Write a program for implementing of Storage classes: (Auto, static, extern, register)

A storage class defines the scope (visibility) and life time of variables and/or functions within a C Program.

There are following storage classes which can be used in a C Program

- **auto** : Formal parameters and local variables of functions are variables that are automatically allocated on the stack when a function is called and automatically deallocated when the function returns.
- **register** :
  Register is a special high-speed memory location inside the central processor
- **static** :
  Static variable is allocated and initialized one time, prior to program execution.
- It remains allocated until the entire program terminates.

- **extern** :
- Storage class of names known to the linker.

  Example:
  - extern int square (int x);
  - Means the function will be available to the linker.
  - It notifies the compiler that such a function exists and that the linker will know where to find it.

VIVA QUESTIONS

1 INTRODUCTION TO C

1) What is an algorithm?

2) What is a flowchart?

3) What are the characteristics of an algorithm?

4) What is a procedural oriented language?

5) How can u say C is a middle-level language?

6) What are the steps required to develop a C program?

7) What are the characteristics of C language?

8) What is difference between compiler and interpreter?

9) What is preprocessor directive?

10) When preprocessing is done?

11) Name some of the activities done at the time of preprocessing?

2 DATA TYPES AND OPERATORS

1) What is a C token?

2) What is a key word? List different keywords in C language?

3) What is a data type? Give different data types in C.

4) What is the use of typedef?

5) How to declare a constant in C?

6) What is the difference between a constant and a variable?

7) Give some examples of header files in C?

8) Why we include header files in a C program?

9) What are the different types of binary operators in C?

10) List different arithmetic operators and what is their precedence?

11) List different relational operators in C.

12) List different logical operators in C and give their precedence.

13) What is conditional operator? Write the syntax.

14) List different unary operators in C.

15) What is the associativity of assignment operator?

16) What is the use of sizeof operator?

20) What are shorthand assignment operators?

21) What is type casting?

22) List different bit-wise operators. What is the use of bit-wise operators?

3 CONTROL STRUCTURES

1) What are the different branch control statements in C language?

2) Is C a block structured language?

3) What is the meaning of block structured language?

4) What are the different loop control statements in C language?

5) What is the difference between while loop and do-while loop?

6) Draw the flowchart of while, do-while and for loop?

7) How many iterations occurs in the following while loop:

```
while(1)   { … }
```

8) How to swap two variables?

9) How to swap two variables without using a temporary variable?

10) What is a compound statement? Give an example.

11) Write the syntax of for loop.

12) How a for loop executes?

13) How many iterations occurs in the following for loop:

```
for( ; ; )   { … }
```

14) How a nested for loop executes?

15) Differentiate between a for loop and a while loop?

16) What is the purpose of break statement in a loop?

17) What is the purpose of continue statement in a loop?

18) What does exit() do?

19) Write the syntax of switch statement.

20) Draw the flowchart of switch statement.

21) In switch control structure when default block is executed?

22) In switch control structure if there is no break statement in particular case block then what happen?

23) Why the usage of goto control structure is not recommended in C?

24) Write the recurrence relation to find nth Fibonacci number.

25) Write the recurrence relation to find the factorial of a given number.

Week 8:

a. Write a C program to find the minimum and maximum integer in a list of integers.

ALGORITHM:

```
Step 1:Start.
Step 2: Read the set of elements
Step 3: min :=max := a [0]
Step 4: for I =1 to n repeat step 5 to 7
Step 5: if a[I]>max      max := a[i]
Step 6: else
Step 7: if a[I] <min     min := a[i]
Step 8: print max
Step 9: print min
Step 10: stop
```

Test Data:

INPUT

```
 Enter how many element : 5
Enter the elements        : 3  4  23  12  43
```

OUTPUT

```
 Minimum =3
Maximum = 43
```

b. Write a C program that uses functions to perform the following:
   i. Addition of Two Matrices
   ii. Multiplication of Two Matrices

ALGORITHM:

```
1. Start
2. Read the operator to be p
3 Using the switch statement for  operator.
   a) Case  `+': call add()
   b) Case  `*': call mull()
4. Stop
```

Procedure add()

```
1.Start
2.Read two  matrices  using  nested  for  loops.
3.Display the two  matrices  using  nested for loops
4.Then add  two matrices using the  following  statement  with  in the
      nested for loop.
C[i][i]= a [i][j]  +b[j][j]
5. Display the resultant matrix using nested for loops.
6. Stop
```

Procedure mull()

```
1.Start
2.Read the value of two  matrices using nested  for loops
3. Display the  two  matrices  using  nested for  loop
4.Then multiply two matrices using the following  statement using nested
   for loop      c[i] [j] = c[i] [j] + a[i] [p] *b[p][j]
5. Display the resultant matrix using nested for loops.
6. Stop
```

Test Data:

INPUT
Enter the choice + or * +
OUTPUT

```
                A=     1    1    1                    B=   1    0     0
                    1    1    1                    0    1     0
                    1    1    1                    0    0     1
```

Resultant Matrix is:

```
2    1    1
1    2    1
1    1    2
```

INPUT: Enter the choice + or *
*

OUTPUT:

```
                A=       1    1    1                  B=   1    0     0
                    1    1    1                    0    1     0
                    1    1    1                    0    0     1
```

Resultant Matrix is:

```
1    1    1
1    1    1
1    1    1
```

iii. Transpose of a matrix

```
1.START
2.Take m1 and m2 as integer matrix.
3.Input the values for original matrix and store it in m1.
4.Convert each row of the matrix m1 in to column of matrix m2.
5.Display both matrix m1 and m2.
6.STOP
```

Week 9:

a. Write a C program that uses functions to perform the following operations:
   - To insert a sub-string in to a given main string from a given position.
   - To delete n Characters from a given position in a given string.

b. Write a C program to determine if the given string is a palindrome or not Inserting a sub – string in to given main string from a given position.

ALGORITHM:

```
Step 1: Start
Step 2: Read a string 'a'
Step 3: Read a sub string 'b'
Step 4: Read a Position from where to be inserted into pos.
Step 5: Compute  string  lengths of main string and substring using function  “strlen()”
and store into m and s respectively
Step 6: If (s>m)  go  to  step  17
Step 7: for i=0 to pos-1 repeat step 8
Step 8: c [i]  = a [i ]
Step 9:  for j = 0 to s    repeat steps 10 and 11
Step 10: c[i]= b[j]
Step 11: i=i+1, j=j+1
Step 12: for j=pos to m repeat steps 13 and 14
Step 13: c[i] = a[ j]
Step 14: i=i+1, j=j+1
Step 15: Add the terminating character to string a
Step 16: a[i]=’\0’
Step 17: display string a
Step 18: stop
```

<u>Test Data:</u>

<u>INPUT</u>

Read the string HEO

Read the sub string LL

Position from where to be inserted 2

<u>OUTPUT</u>

HELLO

ii. Deleting ‘n’ characters from a given position in a given sring.

<u>ALGORITHM:</u>

```
Step 1: Start
Step 2: Read string  form   key board .
Step 3: Read the position from where to delete
Step 4: Read the  no of character to be deleted.
Step 5: call delete  (string, n, pos)
Step 6: stop.
 delete  ( s, n, pos)
     S: string
     n: no of character
     pos: position from where it is to be deleted.
```

```
Step 1: Start
Step 2: compute string length using function "strewn()"
Step 3: Repeat  the loop up to desired position
Step 4: j = pos.
Step 5: Repeat the step 6, 7 till  a [j] is equal NULL
Step 6: a [j+2]
Step 7: j+1
Step 8: stop
```

Test Data:

INPUT

```
        Enter the string
                computer.
    Enter no  of characters to be deleted  3
    Enter the position from which the deletion should be done  2
```

OUTPUT

```
Coter
```

b. Determining the string is palindrome or not

ALGORITHM:

```
Step 1: Start.
Step 2: Read  a string into str
Step 3: Compute string length  using function  "strlen()" and assign to   j
Step 4: i=0, test=0
Step 5: If  (i>j)   go  to step  8.
Step 6: if  (str [i]  is not equal to str[j] )
Step 7: test  = 1
Step 8: i =i+1
Step 9: j = j-1,  go to step 5
Step 10: If  (test  = 0)
Step 11: display  palindrome
Step 12: else
Step 13: display not  a palindrome
Step 14: stop
```

<u>Test Data:</u>

<u>INPUT</u>

Enter a string

MADAM

<u>OUTPUT</u>

palindrome.

Week 10:

a. Write a C program that displays the position or index in the string S where the string T begins, or -1 if S doesn't contain T

<u>procedure:</u>

```
1. Read main string in to s
   Read the substring in to t
2. Use the library function strstr() to find the position of the substring in
   the following way
   i = strstr ( s, t)
4. If i> =0
        print substring starts at position i
   Else
        Print substring does not exit.
```

<u>Test Data:</u>

<u>INPUT</u>

s = Canteen
t = ant

<u>OUTPUT</u>

2

c. Write a C program to count the lines, words and characters in a given text.

<u>procedure:</u>

```
1. Read character in to array 'line' till it is not new line
2. If first character is NULL
   else
3. words = words + 1
4. If line [i] is equal to NULL go to 6 .
5. If (line [i] = ` ' or line[i] = '\t')
6. words = words + 1, go to step 5
7. nline =nline+1
8. compute string length of line and add to characters
```

Test Data:

INPUT

```
                Enter the text :
                        This is computer  Lab
                        We are first years
```

OUTPUT

```
                        no  of lines :        2
                        no  of words :        8
                        no  of characters :  38
```

Week 11:

a. Write a C program to generate Pascal's triangle.

procedure:

```
1.    Input the value  for r
2.    k=1
3.    Repeat the steps  5 to13  until (k<=r)
4.    For  j= 30 –a  to 0 times.
      Print  the  space

5.    For  x = 1 to  k times
6.    If x is equal to 1 or k is equal to 1
      i = 1

7.    else
      i = (i*(k - x +1)) /(x-1)

8.    Print the   value   of   i
9.    x= x+ 1,  go to step  5.
10.   Go to next line on the screen .
11.   k =k+1
12.   a = a+2,   go to step 4
```

Test Data:

INPUT

```
        enter the no. of lines: 4.
```

OUTPUT

```
                              1

                       1     1

                 1           2           1

             1         3           3          1
```

b. To construct a pyramid of numbers .

procedure:

```
1    Input   the value of n.
2    For  i = 1 to n times repeat steps 4 to
4    For  k=1 to n times repeat steps 5 and 6
5 . Print the   space
6   k=k+ 1, go to step4
7.   For  j =1 to i times
        Print  the value  of i
8     Print a new line.
9     j = j+1, go to step7.
10  i = i+1,go to step3
```

Test Data:

INPUT

n = 4

OUTPUT

```
                    1
               2         2

          3    3         3

      4        4         4         4
```

b. Write a C program to construct a pyramid of numbers.

1 1 2 1 2 3	* * * * * *	1 2 3 4 5 6	1 2 2 3 3 3 4 4 4 4	1 1 1 0 1 0 1 0 1 0	* * * * * * * * *

VIVA QUESTIONS

1 ARRAYS

1) What is an array?

2) What is the difference between an array and an ordinary variable?

3) What are the properties of array elements?

4) How to initialize one dimensional array?

5) How to initialize two dimensional array?

6) How a one dimensional array stored in memory?

7) How a two dimensional array stored in memory?

8) How can you increase the size of a dynamically allocated array?

9) How can you increase the size of a statically allocated array?

10) If the following array a is stored in memory using row-major order with 4000 as the starting address, then what is the address of a[3][[4]? The details of the array are

int a[5][4];

size of integer is 2 bytes.

2 FUNCTIONS

1) Define function in C?

2) What are the advantages of functions?

3) What are the different types of functions?

4) Give examples of user defined and pre-defined functions.

5) What is function prototype?

6) What is the difference between actual arguments and formal arguments?

7) What is scope and extent of a variable?

8) What is scope of a variable that is declared in a function?

9) What is the scope of a global variable?

10) How to declare a static variable? What are the features of static variable?

11) What are different types of storage classes in C?

12) What is the difference between macro and ordinary function?

13) What is the default value of static variable?

14) What are register variables?

15) What is the use of extern keyword?

16) What are the different parameter passing mechanisms in C?

17) What is the difference between call by value and call by reference?

18) What is the scope of a global variable, which is declared as static?

19) How can I return multiple values from a function?

20) What is the default value of automatic variable?

3 STRINGS

1) What is a string?

2) What is the difference between Strings and Arrays?

3) How to initialize a string?

4) How to read a line of text?

5) What is the difference between scanf() and getchar() functions?

6) List predefined string handling functions in C.

7) Explain strcmp() function.

8) Explain strcpy(), strlen(), and strrev() functions.

9) What do the functions atoi() and itoa() do?

10) What is the difference between ‘A’ and “A”?

11) How to process array of strings?

12) How to initialize 2Darray of characters?

Week 12:

a. Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression:
1+x+x2+x3+………….+xn
For example: if n is 3 and x is 5, then the program computes 1+5+25+125. Print x, n, the sum
Perform error checking. For example, the formula does not make sense for negative exponents – if n is less than 0. Have your program print an error message if n<0, then go back and read in the next pair of numbers of without computing the sum. Are any values of x also illegal? If so, test for them too.

Procedure:

```
1. Input the value for n and x.
2. If n is less than zero
3 . For i = 0 to n times repeat steps 5
4 . sum = sum + x ^ i
5 . i =i+1, go to step 4
6. Print sum
```

Test Data:

INPUT

n=2 x=3

OUTPUT

15

b. 2’s complement of a number is obtained by scanning it from right to left and complementing all the bits after the first appearance of a 1. Thus 2’s complement of 11100 is 00100. Write a C program to find the 2’s complement of a binary number

procedure:

```
1. Initialize i=0, x=0 , flag = 0,
2. Read a binary number
3. check whether it is a binary number or not.
4. Compute length of string using function "strlen() " and assign
   to x .
5. i=x-1
6. if i<0 go to 10
7. if (flag = 1 )
     a)If a [i] = = `1'
            a [i] = =`0'
     b)else
            a[i] = `1'

8. if (flag=0 and a[i] = `1')
            flag = 1

9. display string a
```

Test Data:

INPUT

Enter a binary no : 1 0 1 1

OUTPUT

2’s Complement of a given no is : 0 1 0 1.

Week 13:

a. Write a functions to compute mean , variance , SD, sorting of N elements in single dimension array.

procedure

```
1. Declare set of elements
2. procedure to mean()
     i. select N elements
    ii. make sum of all elements
   iii. prepare average of N values
3. procedure for Variance()
     i. identify the formula for Variance and utilize mean formula
4. procedure for SD()
   make a squire root of Variance

1. procedure for Sorting() of N elements
2. Declare N elements
3. star loop
4. compare each element with left over elements (use loop)
5. if small element found in that list swap the elements
6. stop loops
```

b. Write a C program to convert a Roman numeral to its decimal equivalent.

ALGORITHM:

```
1. Read a roman number in to string s
 2. [call function validate (s) ]
        valid = validate (s)
3. If valid = 0
        [Call function calculate ]
        value = calculate (s)
 else
      Print 'Invalid number', go to 5
4. print value
```

Procedure validate(s)

```
1. len= length of string s, flag =0
2. for i =0 to len repeat steps 3 to 6
3. If [i] = 'D' and s [ i +1]= 'M'
       flag =1
        [D cannot come to the left of M]
4. If s [i] ='L'
 a) If s[i+1] = 'I' or s[i + 1] ='V'
            flag = 0
     b) else
```

```
                    flag =1
                [L can come to the left of only  I and V]

        5. If  s[i] = 'V' or  s[i] ='L' or s [i] ='D'
                a) If  s [i + 1]   = s [i]
                        flag = 1
                [none of V,  L, D  can be repeated consecutively]
        6. If s[i] ='I'
                If s[i + 1] ='V' or s[i +1] ='X'
                        flag = 0
                else
                        flag = 1
                ['I' can come to the left of only V and X]
        7. Return flag
```

Procedure calculate(s)

```
    1.  sum = 0 , len = length  of s, i = 0
    2.  for  i = 0 to len  repeat steps  3 to 5
    3.  If  s[ i+ 1] = empty
                sum = sum = s [i + 1], return sum
    4.  else if (sum[i]> sum [i+1]
                a) sum = sum + s [i]
                b) i = i+1
                c) go to 2
    5. else if (sum[i]<sum[i+1])
                a)sum = sum + s[i+1] – s[i]
                b) i= i +2
                c) go to 2
```

<u>Test Data:</u>

<u>INPUT</u>

```
        XIV
```

<u>OUTPUT</u>

```
        14
```

Week 14:

a. Write a program for reading elements using pointer into array and display the values using array.
   I. Declare set of elements
   II. Declare pointer and initialize it to first element address of set of elements(array)
   III. Repeat loop until pointer reaching to last element and display each element

b. Write a program for display values reverse order from array using pointer.
   I. Declare set of elements
   II. Declare pointer and initialize it to last element address of set of elements(array)
   III. Repeat loop until pointer reaching to first element and display each element

Write a program through pointer variable to sum of n elements from array .

   I. Declare set of elements
   II. Declare pointer and initialize it to first element address of set of elements(array)
   III. Repeat loop until pointer reaching to nth element and add each element to variable sum
   IV. Display sum

Week 15:

A. Write a C program which copies one file to another.

procedure:

1. Declare a file pointer fp1 in read mode

2. Declare a file pointer fp2 I write mode

3. Read characters of file1 and write to file2

Test Data:

INPUT

OUTPUT

contents of file1 are copied to file2

B. Write a C program to reverse the first n characters in a file. (Note: The file name and n are specified on the command line.)

Procedure:

1. Declare a file pointer fp in read mode
2. Read n (no.of characters to be reversed)
3. Read them into a string
4. Reverse the string
5. Using fseek and putc functions write the characters of the string to the file

Test Data:

INPUT

filename, no.of characters to be reversed

OUTPUT

first n characters of the file are reversed and written back to file

Week 16:

a. Write a C program to display the contents of a file.

procedure:

1. Declare a file pointer fp in read mode
2. Read the characters of file
3. write to characters to stdout.

Write a C program to merge two files into a third file.
(i.e., the contents of the first file followed by those of the second are put in the third file)

procedure:

1. Declare a file pointer fp1 in read mode
2. Declare a file pointer fp2 I read mode
3. Declare a file pointer fp3 I write/append mode
4. Read contents of file1 and write to file3
5. Read contents of file2 and append to file3

Pointers and File's FAQ

1. What is a pointer?
2. Can pointers point to functions?
3. How should we signify that a pointer doesn't point to anything?
4. Can we set one pointer to equal another pointer?
5. What is a void* ?
6. Explain how const works with pointers.
7. what's pointer arithmetic you have mentioned?
8. Can we talk about function pointers now?
9. What is a smart pointer?
10. What is passing by value?
11. what is passing by reference?
12. what is stream ?
13. What is the difference between text and binary modes?
14. Which header file should be included to use functions like malloc() and calloc()?
15. Declare the following statement? "An array of three pointers to chars".

Repeated Questions

1. Write a C program that uses a function to sort an array of integers.
2. What is a pointer? How is a pointer initiated? Give an example.
3. State whether each of the following statements is true or false. Give reasons.
   a. An integer can be added to a pointer.
   b. A pointer can never be subtracted from another pointer.
   c. When an array is passed as an argument to a function, a pointer is passed.
   d. Pointers can not be used as formal parameters in headers to function
   e. definitions.
4. If m and n have been declared as integers and p1 and p2 as pointers to integers, then find out the errors, if any, in the following statements.
   a. p1 = &m;
   b. p2 = n;
   c. m=p2-p1;
   d. *p1 = &n;
   e. Explain the process of accessing a variable through its pointer. Give an Example
5. Explain the process of accessing a variable through its pointer. Give an Example.
6. Write a C program using pointers to read in an array of integers and print its elements in reverse order.
7. Write a program to sort the set of strings in an alphabetical order?
8. How are multidimensional arrays defined? Compare with the manner in which One-dimensional arrays are defined.
9. Explain the process of accessing a variable through its pointer. Give an Example.
10. Write a C program using pointers to read in an array of integers and print its elements in reverse order.
11. Write a C program to read last 'n' characters of the file using appropriate file Function.
12. Write a C program to read a text file and convert the file contents in capital (upper-case) and write the contents in a output file.
13. Write a C program to replace every 5th character of the data file, using fseek() command.
14. Write a program to demonstrate passing an array argument to a function.Consider the problem of finding largest of N numbers defined in an array.

15. Explain the process of declaring and initializing pointers. Give an example.
16. Write a C program that uses a pointer as a function argument.
17. . Explain the process of accessing a variable through its pointer. Give an Example.
18. Write a C program using pointers to read in an array of integers and print its elements in reverse order.
19. . Explain the effects of the following statements:
    1. int a, *b = &a;
    2. int p, *p;
    3. char *s;
    4. a = (float*)&X;

20. Write a C program to do Matrix Multiplications.
21. Write in detail about one dimensional and multidimensional arrays. Also write about how initial values can be specified for each type of array?
22. Explain the process of declaring and initializing pointers. Give an example.
23. Write a 'C' Program to compute the sum of all elements stored in an array using pointers.
24. Write a 'C' program using pointers to determine the length of a character string.
25. In what way array is different from an ordinary variable?
26. What conditions must be satisfied by the entire elements of any given array?
27. What are subscripts? How are they written? What restrictions apply to the Values that can be assigned to subscripts?
28. What advantage is there in defining an array size in terms of a symbolic constant rather than a fixed integer quantity?
29. Write a program to find the largest element in an array.
30. How to use pointers as arguments in a function? Explain through an example.
31. Write a 'C' function using pointers to exchange the values stored in two locations in the memory.
32. How Structure elements are accessed using pointer? Which operator is used? Give an example
33. Write a C program to do Matrix Multiplications.
34. .How are initial values written in a one-dimensional array definition? Is the entire array be initialized? What value is automatically assigned to those array elements not explicitly initialized?
35. Write a program to calculate mean, variance and standard deviation of n
    a. Numbers. S=variance, where
        i. Variance = 1/n sum $(x_i – m)_2$
        ii. M= mean of n numbers.
36. Write a C program using pointer for string comparison.
37. Write a C program to arrange the given numbers in ascending order using Pointers.
38. What are Bit fields? What are its advantages? What is its syntax?
39. Explain the process of accessing a variable through its pointer. Give an Example.
40. Write a C program using pointer for string comparison.
41. Write a C program to arrange the given numbers in ascending order using pointers.
42. Write a C Program to illustrate the use of indirection operator " * " to access the value pointed by a pointer.
43. Write a program to demonstrate passing an array argument to a function.Consider the problem of finding largest of N numbers defined in an array.
44. Write a program to sort the set of strings in an alphabetical order?
45. How are multidimensional arrays defined? Compare with the manner in which one-dimensional arrays are defined.
46. Explain how strings can be stored using a multidimensional arrays?

47. Explain the process of accessing a variable through its pointer. Give an Example.
48. . Write a ‘C’ program to find number of words, blank spaces, special characters,
    a. digits and vowels of a given text using pointers.
49. How are multidimensional arrays defined? Compare with the manner in which one-dimensional arrays are defined.
50. How to use pointer variables in expressions? Explain through examples.
51. Write a ‘C’ Program to illustrate the use of pointers in arithmetic operations.
52. Let a be an array of integers. Present recursive algorithms to compute
    a. Maximum element of the array
    b. The sum of elements of the array
53. What is a pointer? List out the reasons for using pointers.
54. Write a C Program to illustrate the use of indirection operator “*” to access the value pointed by a pointer.
55. How to compare structure variables? Give an example.
56. Write a C program to read a text file and to count
    i. number of characters,
    ii. number of words and
    iii. number of sentences and write in an output file.
57. How does an append mode differs from a write mode.
58. Compare between printf and fprint f functions.
59. Write a program to copy upto 100 characters from a file to an output array.
60. .Distinguish between getchar and scanf functions for reading strings.
61. Write a program to count the number of words, lines and characters in a text.
62. Write a C program to read last ‘n’ characters of the file using appropriate file function.
63. Write a C program to read a text file and convert the file contents in capital (upper-case) and write the contents in a output file.
64. Write a program to count the number of words, lines and characters in a text.
65. Distinguish between the following functions.
    a. Printf and fprintf.
    b. eof and ferror.
66. Write a program to copy the contents of one file into another.
67. What are the file I/O functions in C. Give a brief note about the task per-formed by each function.
68. Write a program to read an input file and count the number of characters in the input file.
69. Distinguish between text mode and binary mode operation of a file.
70. Write a program to open a pre-existing file and add information at the end of file. Display the contents of the file before and after appending.
71. What is the task performed by fseek( ) function. What is its syntax. Explain each parameter in it.

72. Write a C program to read the text file containing some paragraph. Use fseek() and read the text after skipping n characters form beginning of the file.
73. Explain the way of defining, opening and closing a file. Also explain the
    a. different modes of operation.
74. Write a C program to read data from the keyboard, write it to a file called.
75. INPUT, again read the same data from the INPUT file, and display it on the screen.

Outcomes :
After completion of the course, the students would be able to:

- Understand the basic terminology used in computer programming.
- Write, compile and debug programs in C language.
- Design programs involving decision structures, loops and functions.

RFERENCES

WEBSITES:

www.ctutorialspoint.com

www.google.com