Ahsanullah University of Science and Technology (AUST)
Department of Computer Science and Engineering

LABORATORY MANUAL

Course No.: CSE 1288
Course Title: Computer Programming Sessional

For the students of 1st Year, 2nd semester of
B.Sc. in Mechanical Engineering program

TABLE OF CONTENTS

COURSE OUTCOMES

After successful completion of this course, the students should be able to

- Comprehend the basic structure of C program.
- Apply different algorithms to solve various problems using C programming language.
- Analyze complex problems and understand different techniques and debug general programming errors.

PREFFERED TOOL(S)

1. Code Blocks

TEXT/REFERENCE BOOK(S)

1. Herbert Schildt, ***Teach Yourself C***, 3rd Edition.
2. E. Balagurusamy, ***Programming in ANSI C***, 4th Edition.

ADMINISTRATIVE POLICY OF THE LABORATORY

1. Students must perform class assessment tasks individually.
2. Viva will be taken for each assignment and marks on assignment will substantially depend on viva.
3. Plagiarism is strictly forbidden and will be dealt with punishment.

Session 1

1. OBJECTIVES:

i. Getting acquainted with major components of a C program
ii. Typing and Executing a C Program

```
/* Instruction: Observe, Type, Save and Run */
/* A Program to find the sum of a series of integers */

#include <stdio.h>

int main()
{
    int term1, interval, numberOfTerms;
    int sum, lastTerm;

    printf("Enter the first term:");
    scanf("%d", &term1);

    printf("Enter the interval:");
    scanf("%d", &interval);

    printf("Enter the number of terms:");
    scanf("%d", &numberOfTerms);

    lastTerm = term1 + (numberOfTerms - 1)* interval;
    sum = ((term1 + lastTerm) * numberOfTerms) /2;

    printf("The sum is %d.", sum);

    return 0;
}
```

Comment

Header File: stdio.h

Main function headline
Function body begins
Memory reservation or Variable declaration

Output statement
Input statement

Calculating and assigning (remembering) values

Output statement

Function output / Return Statement
Function body ends

Mark the elements of a **function**:

```
int[1] main[2] (void[3])
{
 ……..[4]
……;
……;
.
.
.
……[5]
}
```

1 – Return type

2 – Function name

3 – Argument / parameter list

4 – A statement

5 – Special statement defining function output/ return

Function head

Body of the function as a 'block' of codes or sequence of statements within '{' and '}'.

2. General form of C Programs[Header Files and Library Functions]:

```
/* Instruction: Observe, Type, Compile and Run */
/* A simple multiple function program */

#include <stdio.h>
#include <math.h>

int main()
{
 float  n1, n2, n3;  /* Variable declaration */
 double  n4, n5;

 printf("\nEnter two decimal fractions:");
 scanf("%f %f", &n1, &n2);

 n3 = sum_xy(n1, n2);  /* Function call with Parameters */
 n4 = prdct_xy(n1, n2);
 n5 = sqrt(n4);

 printf("\nThe 1st function returns %f.", n3);
 printf("\nThe 2nd function returns %f.", n4);
 printf("\nThe square root of %f is %f.\n", n4, n5);



 return 0;

}
```

Inclusion of Header files

main function

****** In C programs the main function return type is int and therefore it should return an integer value. The return value of the main function is considered the "Exit Status" of the application. The 0 exit code is a widely accepted convention for 'OK the program execution was successfull'. return 0 means no error.

3. Operators in C Programming:

a. Binary operator: + , - , / , * , %
b. Unary operator: + ,-
c. Relational operator: > , > = ,< , < = , = = , ! =
   [true will return 1 and false will return 0]
d. Logical operator: && (AND) , || (OR) , ! (NOT)
   [true will return a nonzero value and false will return 0]
e. Bitwise operator: & (AND), | (OR), ^ (XOR) , << , >> , ~
f. Increment and decrement operator: + + , - -
g. Assignment operator: = , + = , * = , - = , / = , % = , & = , | = , ^ = , << = , >> =

4. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a program that receives as input through keyboard the measurement of length and breadth of a rectangular piece of land in feet and displays on the monitor the area of the land in square meters, taking 1 ft = 0.3048 m. Arrange input and output with appropriate text messages.

Steps to follow for the solution:

a) Taking input from user and storing the data
b) Conversion of given input from feet to meter
c) Calculation of the area and storing the result
d) Displaying the result

5. EXERCISES:

1. Write a program to input a four-digit integer and display the sum of the digits.
2. Write a C program to take a letter from English alphabet as input and display both the previous and the next letters with ASCII codes. Assume that input will always be chosen from B to Y or b to y.

Sample Input	**Sample Output**
Enter a letter: d	Previous letter with ASCII: **c 99** Next letter with ASCII: **e 101**
Enter a letter: B	Previous letter with ASCII: **A 65** Next letter with ASCII: **C 67**

6. PROBLEMS FOR PRACTICE:

1. Write a C program to enter radius of a circle and find its circumference and area. Note that **Circumference** = 2 x π x radius, **Area** = π x $(\text{radius})^2$, and assume **π**=3.1416

Sample Input	**Sample Output**
Enter radius: 10	Circumference = 62.79 units Area = 314 square units

2. Write a C program to calculate and display the total salary of an employee considering that total salary is the sum of basic salary and house rent. The program must ask the user for the basic salary and percentage of basic salary which determines the house rent.
3. Write a C program that takes number of days as input, and then converts it into years and days, and displays the results. Assume that, 1 year = 365 days.

Sample Input	**Sample Output**
Number of days: 735	Years: 2 Days: 5

4. Write a C program to swap the values of two integer variables with and without using any extra variable.

Session 2

1. OBJECTIVES:

i. Conditional Statements
ii. Relational and Logical Operators

Expression: An expression is the combination of functions, operators, commas, variables, identifiers etc. Ex: x = 0 , x + +.

Statement: An expression as x = 0 , x + + or printf(...) becomes a statement when it is followed by a semicolon(;) as in

```
x + + ;
x = 0 ;
printf(...) ;
```

Braces { and }are syntactically equivalent to a single statement. if ... else: if ... else statement is used to express decisions. Formally the syntax is:

```
if (expression)
        statement1;
else
        statement2;
```

Here the else part is optional. The expression inside if ... else statement expects a value. If the value is nonzero then it is true. If the value is 0 then it is false.

```
/* Instruction: Observe, Type, Compile and Run */
/* A program to find odd or even number */

#include<stdio.h>

int main()
{
        int number;

        printf("Enter a number: ");
        scanf("%d", &number);

        if(number%2==0)
                printf("Number is Even");

        else
                printf("Number is Even");

        return 0;
}
```

else ... if:

```
if(expression 1)
        statement 1
else if(expression 2)
        statement 2
else if(expression 3)
        statement 3
.
.
.
else
        statement n
```

else ... if checks its expression sequentially. When it found one expression is true then it will not check other expressions within that else ... if block.

```
/* Instruction: Observe, Type, Compile and Run */
/* A program to find income tax of an employee according to salary */

#include<stdio.h>

int main()
{
        int salary; float tax;

        printf("Enter your monthly salary: ");
        scanf("%d", &salary);

        if(salary>=9000)
           tax = salary * 0.2;

        else if(salary>=7500 && salary<9000)
                tax = salary * 0.1;

         else if(salary>=6500 && salary<7500)
                tax = salary * 0.05;
        else
                tax = 0;

        printf("Calculated tax = %.2f\n", tax);

        return 0;
}
```

2. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a program to input a three integer number and print them ascending order. [if a user input 9,2,8 then the output of the program will be 2 8 9]

3. EXERCISES:

1. Write a program that takes as input total marks obtained by a student in a course and displays the corresponding letter grade according to the following table.

Marks	Letter Grade
80-100	A
70-79	B
60-69	C
50-59	D
0-49	F

2. Write a program to classify, using nested if-else, a character entry into the following classes:
   A lowercase letter;
   An uppercase letter;
   A digit;
   I don't know.
3. Three numbers are input through keyboard. Write a program to find out the maximum and minimum of these three numbers.
4. Take a year as input and determine whether it is a leap year or not. [**Hint**: Check the input year is divisibility by 4 but not by 100 or by 400]

4. ASSIGNMENT 1:

1. Write a program to find whether a given number is even or odd. If found even, check for divisibility by 4, otherwise check for divisibility by 3.
2. Write a program to input two angles of a triangle and check whether the triangle is a right-angled triangle or not.
3. Write a program to input a letter and display it in opposite case, i.e., if the given letter is in upper case, display it in lower case and vice-versa.
4. According to Gregorian calendar, it was Monday on the date 01/01/1900. If any year is input through the keyboard write a program to find out what is the day on 1st January of this year.

Session 3

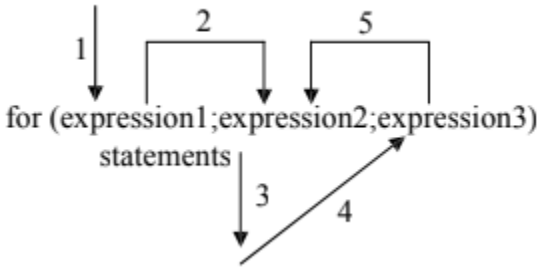
1. OBJECTIVE:

i. Introduction to Loops (for loop)

Loop: Loops are used in C for repeating some portion of the program either a specified number of times or until a particular condition is being satisfied. There are three methods in C for repeating a part of program.

i) Using a for statement
ii) Using a while statement
iii) Using a do ... while statement

for Loop: Syntax of for loop is given below:



Expression1, expression2 and expression3 are normally used for initialization, loop condition and loop counter increment respectively. Expression2 returns a value which may true or false. Any non-zero value means true and zero means false in this case. All three expressions (phase) are optional in for loop.

```
/* Instruction: Observe, Type, Compile and Run */
/* A program to demonstrate for loop */

#include<stdio.h>

int main()
{
        int i;

        for(i=1;i<6;i++)
        {
          printf("\n%d ",i);
          i=i+1;
        }

        for(i=4;i>=1;i=i-1)
        {
           printf("%d ",i)
        }
        return 0;
}
```

2. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a program to find the factorial of a given integer.

3. EXERCISES:

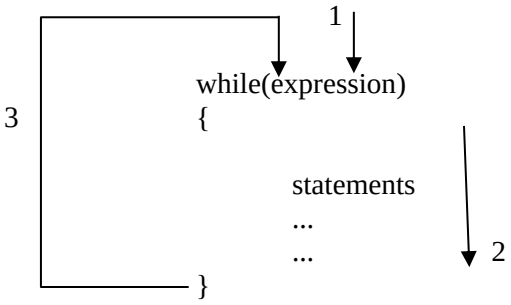
1. Write a program to calculate and display the sum and average of first n odd natural numbers.
2. Write a program to find the sum of the following series up to n terms. Assume entry of a fractional number x.
    i. $x^2/2! + x^3/3! – x^4/4! + \ldots$
    ii. $5 – 11 + 17 - ...$ (up to 75 th term)
    iii. $1 + (1 + 2) + (1 + 2 + 3) + ... + (1 + 2 + 3 + ... + n)$
3. Write a program to determine the GCD (greatest common divisor) and LCM (least common multiple) of 3 numbers.
4. Write a program to determine all prime numbers within the range [a ...b] where a & b are input through keyboard.

Session 4

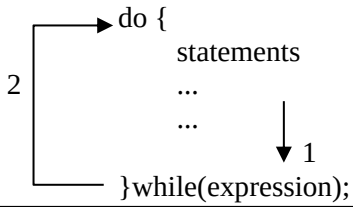
1. OBJECTIVES:

i. Introduction to Loops (while loop and do-while loop)
ii. Practice on Loop

while loop: Syntax of while loop is given below:



do ... while loop: Syntax of do...while loop is given below:




```
/* Instruction: Observe */
/* A program to demonstrate while loop and do-while loop. */

#include <stdio.h>
int main()
{
   int lc=1, check=1, num = 909, guess;

   while(lc<=5 && check)          // check as flag variable
   {
      printf("\nGuess the number:");
      scanf("%d", &guess);
      if(num==guess)
        check=0;
      lc++;
   }
if(check==0)
          printf("\nCongrats! You have guessed in %d tries!\n", --lc);
else
          printf("\nSorry!You failed in all 5 tries.\n");
return 0;}
```

```
/* Instruction: Observe */
/* A program to demonstrate while loop and do-while loop. */

/*using do loop*/
#include <stdio.h>
int main()
{
        int lc=1, check=1, num = 909, guess;

        lc=1, check=1;
        do
        {
                printf("\nGuess the number:");
                scanf("%d", &guess);
                if(num==guess)
                        check=0;
                lc++;
        } while(lc<=5 && check);
        if(check==0)
                printf("\nCongrats! You have guessed in %d tries!\n", --lc);
        else
                printf("\nSorry!You failed in all 5 tries.\n");

        return 0;
}
```

2. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a program to check whether a given integer is a prime number or not.

3. EXERCISES:

1. Write a program to input an integer through the keyboard until the user chooses to quit upon the appearance of options. Every time a number is entered. The program should display whether it is greater than, less than or equal to the previous integer. [Assume initial integer value is 15]

Sample Input and Output
Enter an integer: 23 It is greater than 15. Do you want to continue (y/n)? y Enter an integer: 17 It is less than 23. Do you want to continue (y/n)? y Enter an integer: 17 It is equal to 17. Do you want to continue (y/n)? n

2. Write a program to emulate the **pow** library function where the parameters will be user input.

Sample Input	Sample Output

Enter Base: 3 Enter Power: 5	Result: 243

4. ASSIGNMENT 2:

1. Write a program to find, first using a 'while' loop and then a 'for' loop, the sum of first n terms ($n \geq 1$) of the series 2x3, 3x4, 4x5, …, (n+1)x(n+2). You need to verify that you get the same result in both the cases.
2. Write a program to check whether a given integer is palindrome or not. [*121 is palindrome but 123 is not*]
3. Write a program to print the Fibonacci series up to n terms where n is user input. *[Fibonacci Series: 0, 1, 1, 2, 3, 5, …]*
4. Write a program to print out all Armstrong numbers between 1 and 10000. [Example, 153 = (1*1*1) + (5*5*5) + (3*3*3)].

Session 5

1. OBJECTIVE:

i. Introduction to Nested Loop

```
/* Instruction: Observe */
/* A program to demonstrate a nested 'for' loop */
 #include <stdio.h>
int main()
{
   int i, j, t, p;
   char k;

   printf("\nHow many times? ");
   scanf("%d", &t);

   printf("\nHow many pairs each time? ");
   scanf("%d", &p);

   for(i=1; i<=t; ++i)
   {
                printf("\n\nEnter a character:");
                getchar();
                scanf("%c",&k);

                for(j=0; j<p; ++j)
                        printf("%c\t%c\n", k+j,k+j+1);
   }

   return 0;
}
```

2. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a program to print the following series up to n terms where n is a user input.

(1) + (1+2) +(1+2+3) + (1+2+3+4) + (1+2+3+4+5) + ……..

3. EXERCISES:

1. Write a program to print all the prime numbers between two given integers using ‘for’ and ‘while’ as outer and inner loop respectively.
2. Write separate programs to print the following patterns taking number of rows as user input.

(a)	(b)
* * * * * * * * * * * * * * *	a a ab bb abc ccc abcd dddd abcde eeeee

4. ASSIGNMENT 3:

1. Write a program to determine all Pythagorean triplets without duplication and with minimum possible computational cost in the range 1 to 1000.
   (A Pythagorean triplet is a set of three integers i, j, k such that $i^2 + j^2 = k^2$. Use ‘break’ and/or ‘continue’ statements, where possible.)
2. Write a C program to remove white spaces from a given string.
3. Write a program that will read an integer between 20 and 99 from user and print it out in words. You have to use **switch** statement and repeat the whole process ‘n’ [user input] times. Note that, you can use no more than 2 switch statements with 10 cases per switch at most.

Sample Input	Sample Output
Enter an integer [20-99]: 38	In words: Thirty Eight
Enter an integer [20-99]: 23	In words: Twenty Three

4. Construct the following table. Here n is input from the user.

1	2	3	...	n
2	4	6	...	2n
3	6	9	...	3n
. .	.		...	.
. .	.		...	.
. .	.		...	.
n	2n	3n	...	nn

Session 6

1. OBJECTIVES:

i. Introduction to Multiple Function Program

```
/*Instruction: Observe*/
/*A program to demonstrate simple use of 'global' & 'local' variable and function */

#include<stdio.h>

char char_f1();
void odd_even(int);

int count;

int main()
{
        int num1;
        char ch1;
        count = 1;

        ch1 = char_f1();

        num1 = ch1;
        printf("\nYou have entered %c,", ch1);
        printf(" the code of which is %d.\n", num1);

        odd_even(num1);

        printf("\nValue of global \'count\': %d", count);
        return 0;
}

char char_f1()
{
        char ch2;   int count;
        printf("\nEnter a character:");
        scanf("%c", &ch2);

        count = ch2;
        printf("\nValue of local \'count\' in char_f1(): %d", count);
        return ch2;
}

void odd_even(int n)
{
        if (n % 2 == 0)
          printf("\nIt is even.\n");
        else
          printf("\nIt is odd.\n");

        ++count;
}
```

```
/* Instruction: Observe */
/* A program to demonstrate simple use static variables and function.  */
#include <stdio.h>
int add_amount(int new_amount);
int main()
{
   int n1, n2;
   printf("\nEnter a positive integer:");
   scanf("%d", &n1);
   n2=add_amount(n1);
   printf("\nThe total amount now: %d.\n", n2);
   printf("\nEnter a positive integer:");
   scanf("%d", &n1);
   n2=add_amount(n1);
   printf("\nThe total amount now: %d.\n", n2);
   return 0;
}
int add_amount(int new_amount)
{
   static int total_amount =50; /*Declaration & initialization of static variable*/
   total_amount = total_amount+ new_amount;
   return total_amount;
}
```

2. EXERCISES:

1. Write a multifunction program to print the following patterns where number of rows is user input and must be read in main function. There should be separate function for each of the following patterns and note that, you cannot pass any data through parameters to those functions.

(a)	(b)	(c)	(d)
4444444 33333 222 1	1 2 2 3 3 4 4 555555555	7654321 54321 321 1	* * * * * * * * * * * * * * * * *

2. Write a function to calculate the factorial value of any integer entered through the keyboard.
3. A prime integer is entered through the keyboard. Write a function to obtain the prime factors of this number. For example, prime factors of 24 are 2, 2, 2 and 3 whereas prime factor of 35 are 5 and 7.

Session 7

1. OBJECTIVE:

i. Introduction to Recursion and Recursive Function

A function is called recursive if a statement within the body of a function calls the same function. Sometimes called circular definition, recursion is thus the process of defining something in terms of itself.

```
/* Instruction: Observe */
/* A program to demonstrate simple use recursive functions. */

#include <stdio.h>

int fctrl(int x);

int main()
{
   int i, n1, n2, f1;

   printf("\nEnter a positive integer:");
   scanf("%d", &n1);

   f1=fctrl(n1);        /* Factorial using recursion */

   printf("\nThe factorial of %d is %d.\n", n1, f1);

   return 0;
}

int fctrl(int n)
{
    if (n<2)
        return 1;
    else
        return n * fctrl(n-1);
}
```

2. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a program to find the sum of a series of positive even numbers using recursion.

3. EXERCISES:

1. Write a program to find the sum of a series of positive odd numbers using recursion.
2. The series 0, 1, 1, 2, 3, 5, 8, 13, … is called the Fibonacci series. Here, $term_n = term_{n-1} + term_{n-2}$, for $n>1$, $term_0 = 0$, $term_1 = 1$. Write a program that finds the sum of first n terms of the series using recursion.

3. Convert a decimal number into correspondent binary number using recursion where decimal number is input from user.
4. A positive integer is entered through the keyboard, write a program to obtain the prime factors of the number. Modify the function suitability to obtain the prime factors recursively.

4. ASSIGNMENT 4:

1. Write a multifunction program to print the following patterns where number of rows is user input and must be read in main function. There should be separate function for each of the following patterns and note that, you cannot pass any data through parameters to those functions.

(a)	(b)
7654321 54321 321 1	* * * * * * * * * * * * * * * * *

2. The total number of moves of disks required to solve the Tower of Hanoi problem with n disks is expressed in the following equation:

   $H_n = 2H_{n-1} + 1$, for $n>1$ and $H_1 = 1$.

   Write a program that finds the total number of moves for any given number n of disks using global variables and recursion. Write two different functions that are called from the main function.

Session 8

1. OBJECTIVE:

i. Introduction to One Dimensional Array

```
/* Instruction: Observe */
/* A program to demonstrate inserting and retrieving data in 1-D array */
#include <stdio.h>
int main()
{
   int i, a1[3], s;
   for(i=0; i<3; ++i)
      a1[i] = i+10;
   printf("\nContents of array a1: ");
   for(i=0; i<3; ++i)
     printf("%d ", a1[i]);
   for(i=0; i<3; ++i)
   {
      printf("\nEnter an integer:");
      scanf("%d", &a1[i]);
   }
   printf("\nNew contents of array a1: ");
   for(i=0, s=0; i<3; ++i)
   {
      printf("%d ", a1[i]);
       s=s+a1[i];
   }
   printf("\nThe sum of the elements of array a1:%d ", s);
   return 0;
}
```

2. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a program to read 10 characters from user into an array but display only those characters with ASCII value more than 85. Then ask the user again to enter a character which will be searched for in that array and display "Found" or "Not Found" on the screen.

Sample Input and Output
Enter 10 characters: A1a2"3%0(Z Output: a Z Enter a character to search for: A Output: Found.

3. EXERCISES:

1. Write a program to find the maximum, minimum and average from a list of floating point numbers.
2. Write a program to make a list of integers taking input from user. Then insert a value in that list at a desired location (*not index*) which will be user input as well.

Sample Input and Output
Enter 5 integers: 2 5 0 9 1 Enter data to insert in the list: 33 Enter location to insert at: 3 List before insertion of new data: 2 5 0 9 1 List after insertion of new data: 2 5 33 0 9 1

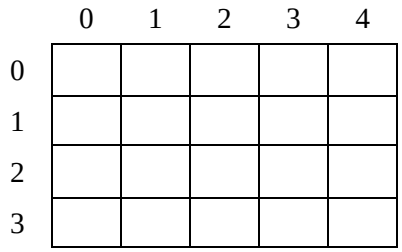
3. Find k-th maximum and k-th minimum from an array.
4. Write a program to delete duplicate elements from an array.
5. Write a program to merge two sorted arrays.

Session 9

1. OBJECTIVE:

ii. Introduction to Two-Dimensional Array

Two dimensional arrays: Say, int a[4][5]




```
/* Instruction: Observe */
/* A program to demonstrate inserting and retrieving data in 2-D array */

#include <stdio.h>
int main(void)
{
   int a[3][4], i, j ;

   printf("Enter the elements of the array:")'

   for( i = 0; i < 3 ; i ++)
       for( j = 0 ; j < 4 ; j ++)
              scanf("%d", &a[i][j]);

  printf("Elements are: ");

  for( i = 0; i < 3 ; i ++)
 {
      for( j = 0 ; j < 4 ; j ++)
             printf("%d", a[i][j] );
     printf("\n");
  }
}
```

2. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a program to read two matrix from user into two different 2D array and display the sum of the matrix.

3. EXERCISES:

Problem: Write a program to read two matrixes from user into two different 2D array and multiply these two matrixes and finally display the result.
Problem: Write a program to display the numbers divisible by 4 or 7 from a 2D array of integers and find the sum of those numbers as well.

Session 10

1. OBJECTIVE:

i. Introduction to Array of Characters and String

```
/* Instruction: Observe */
/* A program to demonstrate concepts of array of characters and string */

#include<stdio.h>
#include<conio.h>
#include<string.h>

int main()
{
   char chArr[13]={'T','h','i','s',' ','i','s',' ','A','U','S','T','.'};
   int i;

   printf("Contents of chArr:\n");
   for(i=0; i<13; i++)
      printf("%c",chArr[i]);

   printf("\n\n");
   printf("Enter new characters for chArr:\n");
   for(i=0; i<13; i++)
      chArr[i]=getche();

   printf("\nContents of chArr:\n");
   for(i=0; i<13; i++)
      printf("%c",chArr[i]);

   char str[13]={'T','h','i','s',' ','i','s',' ','C','S','E','.','\0'};
   printf("\n\nPrinting str as string: ");
   printf(str);

   printf("\n\nEnter a string: ");
   gets(str);
   printf("\nPrinting str char by char: ");
   for(i=0; str[i]!='\0'; i++)
      printf("%c",str[i]);

   return 0;
}
```

2. LET'S WRITE A PROGRAM TOGETHER:

Problem: Write a C program that takes two strings as input and displays the concatenated string putting the lengthier one at the end in uppercase without using **strcat** library function.

Sample Input	**Sample Output**
Enter 1st string: Department of Enter 2nd string: Computer Science	Concatenated String: "Department of COMPUTER SCIENCE"

3. EXERCISES:

1. Write a C program to make a copy of a given string.
2. Write a C program to find the number or vowels, consonants, digits and words in a string.

4. PROBLEM FOR PRACTICE:

```
/*A program to demonstrate manipulation on strings*/

#include <stdio.h>
#include <string.h>
#include <conio.h>

int main()
{
 char c1, str1[30], str2[30];     int i;

 printf("\nEnter a string ( <30 characters): ");
 gets(str1);  printf(str1);

 printf("\nEnter another string: ");
 for(i=0; i<29; ++i)
 {
    c1 = getche();
    if (c1 == '\r') break;
    str2[i] = c1;
 }
 str2[i] = '\0';

 printf("\n\nThe 1st string: "); puts(str1);

 printf("\nThe 1st string, again: ");
 for(i=0; i<29; ++i)
 {
    if (str1[i] == '\0') break;
    printf("%c",str1[i]);
 }
 printf("\nThe 2nd string: "); puts(str2);
 printf("\nLength of 1st string: %d", strlen(str1));
 printf("\nLength of 2nd string: %d", strlen(str2));

 if (!strcmp(str1, str2))
    printf("\nExactly the same strings!\n");
 else
    printf("\nNot the same strings!\n");

 strcat(str1, str2);
 printf("\nThe strings, concatenated: ");
 puts(strupr(str1));

 strcpy(str1, str2);
 printf("\nThe 2nd string, again: ");
 puts(strlwr(str1));

printf("\nEnter character to search in 2nd string: ");
 c1 = getche();
 if (strchr(str2,c1))
   printf("\n%c is in %s.", c1, str1);
 else
   printf("\n%c is not in %s.", c1, str1);
 printf("\nEnter string to search in 2nd string: ");
 gets(str1);
 if (strstr(str2, str1))
    printf("\n%s is in %s.", str1, str2);
 else
    printf("\n%s is not in %s.", str1, str2);
  return 0;
}
```

Session 11

1. OBJECTIVES:

i. Introduction to Structure and Array of Structures

```
/* Instruction: Observe */
/* A program to demonstrate simple use of array of structures. */

#include<stdio.h>
#include<string.h>

struct mystrct
{
   char name[30];
   int age;
   float cgpa;
} student[3];

int main()
{
   int i;
   for(i=0; i<3; i++)
   {
      printf("\n\n::: Student %d :::\n",i+1);
      printf("\nName: ");       gets(student[i].name);
      printf("Age: ");          scanf("%d",&student[i].age);
      printf("CGPA: ");         scanf("%f",&student[i].cgpa);
      getchar();
   }

   printf("\n::: Here are the students' info  :::\n");
   for(i=0; i<3; i++)
   {
      printf("\n\n::: Student %d :::\n",i+1);
      printf("\nName: ");
      puts(student[i].name);
      printf("Age: %d",student[i].age);
      printf("\nCGPA: %f\n",student[i].cgpa);
    }

   return 0;
}
```

**Rewrite this program for 'n' students where n will be user input.

2. EXERCISES:

1. Write a program to input details (name, id and CGPA) of 10 students and display only those student's information whose CGPA is greater than a given range from user.

3. ASSESSMENT:

- ✓ Submission and Viva of Assignment-4.

Session 12

1. OBJECTIVE:

i. Introduction to Pointer

```
/*Instruction: Observe, Type and Run*/
/*A program to demonstrate basic pointer operation. */

#include <stdio.h>

int main()
{
        int x = 2, y =3, * p, * q ;
        p = & x ;
        q = & y ;
        p = q ;
        printf("%d %d %d %d", x, y, * p, * q );
        * p = 3;
        * q = 4;
        x = y;
        printf("\n%d %d %d %d", x, y, * p, * q );
}
```

ii. Pointer arithmetic operation

```
/* Instruction: Observe, Type and Run */
/* A program to demonstrate increment operation with pointers. */

#include<stdio.h>

int main()
{
   char *a, a1; int *b, b1; float *c, c1; double *d, d1;
   a = &a1;    b = &b1;    c = &c1;    d = &d1;

   printf("\ns_a = %d  s_b = %d  s_c = %d  s_d = %d\n", a, b, c, d);
   printf("\nAfter incrementation:");
   printf("\ns_a = %d  s_b = %d  s_c = %d  s_d = %d\n",++a,++b,++c,++d);

   return 0;
}
```

iii. Function Call using reference(call by reference) / pointer

```
/*Instruction: Observe, Type and Run*/
/*A program to demonstrate passing arrays to a function through pointers. */

#include <stdio.h>
void dsplsum(int *i, int j);

int main()
{
  int num[]={100, 200, 300, 1000, 2000, 3000, 4000, 5000};
  dsplsum(&num[0], 5);
  return 0;
}

void dsplsum(int *i, int j)
{
   int sum = 0, k;
   for(k=0; k<j; ++k, ++i)
      sum += *i;
   printf("\nThe sum of the 1st %d elements is %d.", j, sum);
}
```

iv. Dynamic memory allocation using pointer.

```
/* Instruction: Observe */
/* A program to demonstrate inserting and retrieving data in 2-D array using dynamic
memory allocation */

#include <stdio.h>
int main()
{
        int i, j, row, col, *p;

        scanf("%d%d",&row,&col);

        p = (int *) malloc ( row * col * sizeof(int));

        for( i = 0; i < row ; i ++)
                for( j = 0 ; j < col ; j ++)
                        scanf("%d", (p + i * col + j ) );

        for( i = 0; i < row ; i ++)
        {
                for( j = 0 ; j < col ; j ++)
                        printf("%4d", *(p + i * col + j ) );
                printf("\n");
        }
        return 0;
}
```

Session 13

2. OBJECTIVE:

i. Introduction to File and Pointer

```
/* Instruction: Observe, Type and Run */
/* A program to demonstrate basic operations on file. */

#include <stdio.h>
#include <stdlib.h>

int main()
{
   FILE *fp;
   char c; inti;

   printf("\nEnter text (type * and press enter to finish): ");
   fp = fopen("textfile1.txt", "w");

   while((c = getchar()) != '*')
      fputc(c, fp);
   fclose(fp);

   printf("\n\nHere is your text:\n\n");
   fp = fopen("textfile1.txt","r");

   while((c = fgetc(fp)) != EOF)
      printf("%c", c);
   fclose(fp);

   return 0;
}
```

ii. PROBLEM FOR PRACTICE:

```
** A program to maintain a data file.
   #include <stdio.h>
   #include <string.h>
   #include <stdlib.h>

   int int_scn();
   void create_fl();
   void add_to_fl();
   void dspl_items();

   struct mystrct
   {
         char name[30], occupation[15];
         int age;
   } strct1;
```

```
FILE *fp;
int main()
{
  char c1;

  do
  {
       printf("\n1. Create Data File \n2. Add Items  \n3. Display Items \n4. Exit");
       printf("\n\nEnter your choice:");
       c1 = getche();

       if (c1 <'1' && c1 >'4')  continue;

       switch(c1)
       {
         case '1': create_fl(); break;
         case '2': add_to_fl(); break;
         case '3': dspl_items(); break;
         case '4': printf("\n\nConfirm exit by pressing 4 again!\n");
                   c1 = getche();
       }
  } while (c1 != '4');

  return 0;
}

void create_fl()
{
   fp = fopen("strct_file.txt", "w");
   fclose(fp);
   printf("\nDone. Press any key!");
   getch();
}

int int_scn()
{
   char c1, num[15]; int i;

   for(i=0; i<15; ++i)
   {
      c1 = getche();
      if (c1 == '\r') break;
      num[i] = c1;
   }
   num[i] = '\0';

   i = atoi(num);

   return i;
}
void add_to_fl()
{
   char c, csn[30], ocp[15];
   int flag = 1, a;

   fp = fopen("strct_file.txt", "a");
```

```
        while(flag)
        {
          printf("\n\nEnter y to add entry and n to end: ");
          c = getche();
          if (c=='y' || c=='Y')
          {
            printf("\n\nEnter the name:");
            gets(csn);
            strcpy(strct1.name, csn);
            strcat(strct1.name,"\n\0");
            printf("Name the occupation:");
            gets(ocp);
            strcpy(strct1.occupation, ocp);
            strcat(strct1.occupation,"\n\0");
            printf("Enter the age:");
            a = int_scn();
            strct1.age = a;
            fputs(strct1.name, fp);
            fputs(strct1.occupation, fp);
            fprintf(fp, "%d", strct1.age);
          }
          else
            flag = 0;
         }
         fclose(fp);
      }

      void dspl_items()
      {
        char c;
        fp = fopen("strct_file.txt", "r");
        while(!feof(fp))
        {
          printf("\n\nEnter y to display a record and n to end: ");
          c = getche();
          if (c=='y' || c=='Y')
          {
            fgets(strct1.name, 30, fp);
            fgets(strct1.occupation, 15, fp);
            fscanf(fp, "%d", &strct1.age);
            printf("\n\nName: %s\nOccupation: %s\nAge: %d\n\n",
              strct1.name, strct1.occupation, strct1.age);
          }
          else
            break;
        }
        fclose(fp);
}
```

MID TERM EXAMINATION

There will be a 40-minutes written mid-term examination. Different types of questions will be included such as MCQ, mathematics, writing code fragments etc.

FINAL TERM EXAMINATION

There will be a one-hour written examination. Different types of questions will be included such as MCQ, mathematics, write a program etc.