Parul University

FACULTY OF ENGINEERING & TECHNOLOGY

BACHELOR OF TECHNOLOGY

Computational Thinking for Structure Design -1
(303105104)

1st SEMESTER

COMPUTER SCIENCE & ENGINEERING DEPARTMENT

Laboratory Manual

COMPUTATIONAL THINKING FOR STRUCTURE DESIGN -1 PRACTICAL BOOK

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

PREFACE

It gives us immense pleasure to present the first edition of ***Computational Thinking for Structure Design -1*** for the B.Tech. 1st year students for PARUL UNIVERSITY.

The Fundamental of Programming theory and laboratory courses at **PARUL UNIVERSITY, WAGHODIA, VADODARA** are designed in such a way that students develop the basic understanding of the subject in the theory classes and then try their hands on the computer learnt during the theoretical sessions.

This book is emphatically not focused on "the syntax of C". Understanding the fundamental ideals, principals, and techniques is the essence of a good programmer. Only well-designed code has a chance of becoming part of a correct, reliable, and maintainable system. Also, "the fundamentals" are what last: they will still be essential after today's language and tools have evolved or been replaced.

We acknowledge the authors and publishers of all the books which we have consulted while developing this Practical book. Hopefully this *Computational Thinking for Structure Design -1* will serve the purpose for which it has been developed.

Instructions to students

1. Every student should obtain a copy of laboratory Manual.
2. Dress Code: Students must come to the laboratory wearing.
   i. Trousers,
   ii. half-sleeve tops and
   iii. Leather shoes. Half pants, loosely hanging garments and slippers are not allowed.
3. To avoid injury, the student must take the permission of the laboratory staff before handling any machine.
4. Students must ensure that their work areas are clean and dry to avoid slipping.
5. Do not eat or drink in the laboratory.
6. Do not remove anything from the computer laboratory without permission.
7. Do not touch, connect or disconnect any plug or cable without your lecturer/laboratory technician's permission.
8. All students need to perform the practical/program.

CERTIFICATE

This is to certify that

Mr./Ms.…….. with enrolment no. ..... has successfully completed his/her laboratory experiments in the ***Computational Thinking for Structure Design - 1 (303105104)*** *from the department of ..... during the academic year .....*

Date of Submission:..... Staff In charge:.....

Head of Department:.....

INDEX

Class: 1st Sem **Subject: - Computational Thinking for Structured Design-1**
A.Y. 20__-20__ **Subject Code: 303105104**

Sr. No.	Experiment Title	Page No.		Date of Performance	Date of Assessment	Marks out 0f 10	Sign
	Practical Set – 1	**To**	**From**				
1	Installation C IDE, Basic Structure of C program.Format Specifiers, Escape Character. Run time input/Output Programs.						
2	1. Write a c program to calculate Area of Rectangle,Perimeter of a Rectangle and Diagonal of a Rectangle. 2. Write a c program to calculate Area of square,Perimeter of a square and Diagonal of a square. 3. Write a c program to calculate total area ofCylinder and volume of a cylinder.						
3	1. The total distance traveled by vehicle in ⍰t seconds is given by distance s = ut+1/2at? where ‘u’ and ‘a’ are the initial velocity (m/sec.) and acceleration(m/sec?). Write a C program to find the distance traveled 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’. 2. 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)						
4	1. Write a C program to find the sum of individual digits of a positive integer.						

	2. A Fibonacci sequence is defined as follows: the first and second terms in the sequences 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. 3. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.						
5	1. Write a C program to calculate the following Sum: Sum=1-x^2/2! +x^4/4!-x^6/6!+x^8/8!-x^10/10!. 2. Write a C program to find the roots of a quadratic equation.						
6	Write C programs that use both recursive and non-recursive functions. 1. To find the factorial of a given integer. 2. To find the GCD (greatest common divisor) of two given integers.						
7	1. Write a C program to find the largest integer in a list of integers, 2. Write a C program that uses functions to perform the following: 1. Addition of Two Matrices 2. Multiplication of Two Matrices						
8	1. Write a C program that uses functions to perform the following operation; 1. To insert a sub-string into a given main string from a given position. 2. To delete n Characters from a given position in a given string, 3. Write a C program to determine if the given string is a palindrome or not.						
9	1. 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. **2.** Write a C program to count the lines, words and characters in a given text.						
10	1. Write a C program to generate Pascal's triangle. 2. Write a C program to construct a pyramid of numbers.						
11	Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression: 1+x+x^2+x^3+... x^n. 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 without computing the sum. Are any values of x also illegal? If so, test for them too.						
12	1. 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. 2. Write a C program to convert a Roman numeral to its decimal Equivalent.						
13	1. Write a c program on Given an unsorted array arr[] of size N. Rotate the array to the left (counter-clockwise direction) by D steps, where D is a positive integer. 2. Write a c Program on given two sorted arrays arr1 and arr2 of size N and M respectively and an element K. The task is to find the element that would be at the k'th position of the final sorted array.Explanation: Input : Array 1 - 1 4 2 3 5 Array 2 - 7 8 6 k = 5 Output : 5 Because The final sorted array would be -1, 2, 3, 4, 5, 6, 7, 8, The 5th element of this array is 6.						
14	1. Write a c program to take multiline string input and print individual string length . 2. Write a c program to reverse the individual word of a given string Explanation:input : Welcome To Bytexl output: emocleW oT lxetyB.						

1. Installation C IDE, Basic Structure of C program. Format Specifiers, Escape Character. Run time input/Output Programs.

The basic structure of a C program is divided into 6 parts which makes it easy to read, modify, document, and understand in a particular format. C program must follow the below-mentioned outline in order to successfully compile and execute. Debugging is easier in a well-structured C program.

Sections of the C Program
There are 6 basic sections responsible for the proper execution of a program. Sections are mentioned below:

a. Documentation
b. Preprocessor Section
c. Definition
d. Global Declaration
e. Main() Function
f. Sub Programs

1. Documentation
This section consists of the description of the program, the name of the program, and the creation date and time of the program. It is specified at the start of the program in the form of comments. Documentation can be represented as:

```
// description, name of the program, programmer name, date, time etc.
/*
   description, name of the program, programmer name, date, time etc.
*/
```

Anything written as comments will be treated as documentation of the program and this will not interfere with the given code. Basically, it gives an overview to the reader of the program.

2. Preprocessor Section
All the header files of the program will be declared in the preprocessor section of the program. Header files help us to access other's improved code into our code. A copy of these multiple files is inserted into our program before the process of compilation.

Example:

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

3. Definition

Preprocessors are the programs that process our source code before the process of compilation. There are multiple steps which are involved in the writing and execution of the program. Preprocessor directives start with the '#' symbol. The #define preprocessor is used to create a constant throughout the program. Whenever this name is encountered by the compiler, it is replaced by the actual piece of defined code.
Example:

```
#define long long ll
```

4. Global Declaration
The global declaration section contains global variables, function declaration, and static variables. Variables and functions which are declared in this scope can be used anywhere in the program.

Example:

```
int num = 18;
```

5. Main() Function
Every C program must have a main function. The main() function of the program is written in this section. Operations like declaration and execution are performed inside the curly braces of the main program. The return type of the main() function can be int as well as void too. void() main tells the compiler that the program will not return any value. The int main() tells the compiler that the program will return an integer value.

Example:

```
void main()
or
int main()
```

6. Sub Programs
User-defined functions are called in this section of the program. The control of the program is shifted to the called function whenever they are called from the main or outside the main() function. These are specified as per the requirements of the programmer.

Example:

```
int sum(int x, int y)
{
   return x+y;
}
```

List of Format Specifiers in C

The below table contains the most commonly used format specifiers in C

Format Specifier	Description
%c	For b type.
%d	For signed integer type.
%e or %E	For scientific notation of floats.
%f	For float type.
%g or %G	For float type with the current precision.
%i	Unsigned integer
%ld or %li	Long
%lf	Double
%Lf	Long double
%lu	Unsigned int or unsigned long
%lli or %lld	Long long
%llu	Unsigned long long
%o	Octal representation
%p	Pointer
%s	String
%u	Unsigned int
%x or %X	Hexadecimal representation

Format Specifier	Description
%n	Prints nothing
%%	Prints % character

Escape Sequence List

The table below lists some common escape sequences in C language.

Escape Sequence	Name	Description
\a	Alarm or Beep	It is used to generate a bell sound in the C program.
\b	Backspace	It is used to move the cursor one place backward.
\f	Form Feed	It is used to move the cursor to the start of the next logical page.
\n	New Line	It moves the cursor to the start of the next line.
\r	Carriage Return	It moves the cursor to the start of the current line.
\t	Horizontal Tab	It inserts some whitespace to the left of the cursor and moves the cursor accordingly.
\v	Vertical Tab	It is used to insert vertical space.
\\	Backlash	Use to insert backslash character.
\’	Single Quote	It is used to display a single quotation mark.
\”	Double Quote	It is used to display double quotation marks.
\?	Question Mark	It is used to display a question mark.
\ooo	Octal Number	It is used to represent an octal number.
\xhh	Hexadecimal	It represents the hexadecimal number.

Escape Sequence	Name	Description
	Number	
\0	NULL	It represents the NULL character.

Example:

```
#include <stdio.h>
int main() {
  // printf() displays the string inside quotation
  printf("Hello, World!");
  return 0;
}
```

Practical 2

1. Write a c program to calculate Area of Rectangle, Perimeter of a Rectangle and Diagonal of a Rectangle.

```
#include<stdio.h>

#include<math.h>

int main(){

        float a,p,d,l,b;

        /*

        *a=Area

        *p=Perimeter

        *d=Diagonal

        *l=Length

        *b=Breath

        */

        printf("Enter the Length of Rectangle = ");

        scanf("%f",&l);

        printf("Enter the Bareth of Rectangle = ");

        scanf("%f",&b);

        a=l*b;

        printf("Area of Rectangle is %f\n",a);

        p=2*(l+b);

        printf("Perimeter of Rectangle is %f\n",p);

        d=sqrt(l*l+b*b);

        printf("Diagonal of Rectangle is %f",d);

        return 0;

}
```

Output:

Enter the Length of Rectangle = 10
Enter the Bareth of Rectangle = 20
Area of Rectangle is 200.000000
Perimeter of Rectangle is 60.000000
Diagonal of Rectangle is 22.360680

2. Write a c program to calculate Area of square, Perimeter of a square.

```
#include<stdio.h>
int main(){
        float a,p,l;
        /*
        *a=Area
        *p=Parameter
        *l=Length
        */
        printf("Enter the Length of Square = ");
        scanf("%f",&l);
        a=l*l;
        printf("Area of Square is %f\n",a);
        p=4*l;
        printf("Parimeter of Rectriangle is %f\n",p);
        return 0;
}
```

Output:

Enter the Length of Square = 10

Area of Square is 100.000000

Perimeter of Square is 40.000000

3. Write a c program to calculate total area of Cylinder and volume of a cylinder.

```
#include<stdio.h>
int main(){
        float a,v,r,h;
        /*
        *a=Area
        *v=Volume
        *r=Radius
        *h=Height
        */
        printf("Enter the Radius of Cylinder = ");
        scanf("%f",&r);
        printf("Enter the Height of Cylinder = ");
        scanf("%f",&h);
        a=2*(22/7*r*(r+h));
        printf("Area of Cylinder is %f\n",a);
        v=(22/7*r*r*h);
        printf("Volume of the Cylinder is %f\n",v);
        return 0;
}
```

Output:

Enter the Radius of Cylinder = 10
Enter the Height of Cylinder = 50
Area of Cylinder is 3600.000000
Volume of the Cylinder is 15000.000000

Practical 3

1. The total distance traveled by vehicle in 't' seconds is given by distance $s = ut+1/2at^2$ where 'u' and 'a' are the initial velocity (m/sec.) and acceleration(m/sec^2). Write a C program to find the distance traveled 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'.

```
#include<stdio.h>

#include<conio.h>

#include<math.h>

int main(){

        int i;

        float s,u,a,t,t1,t2;

        /*

        *s = Distance

        *u = Initial Velocity

        *a = Acceleration

        *t = Time

        *t1 = First Interval

        *t2 = Last Interval

        */

        printf("Enter Initial Velocity, u = ");

        scanf("%f",&u);

        printf("Enter Acceleration, a = ");

        scanf("%f",&a);

        printf("Enter Time, t = ");

        scanf("%f",&t);

        printf("Enter First Interval,t1 = ");
```

```
        scanf("%f",&t1);
        printf("Enter Last Interval,t2 = ");
        scanf("%f",&t2);
        for(i=t1;i<=t2;i=i+t1){
                s=(u*i)+(0.5*a*i*i);
                printf("Distance Calculated in time %d is %f\n",i,s);
        }
        return 0;
}
```

Output:

Enter Initial Velocity, u = 10

Enter Acceleration, a = 5

Enter Time, t = 10

Enter First Interval, t1 = 1

Enter Last Interval, t2 = 15

Distance Calculated in time 1 is 12.500000

Distance Calculated in time 11 is 412.500000

2. 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)

```
#include<stdio.h>

int main()

{

   int a, b;

   char choice;

   printf("Enter your choice\n");

   printf("a. Addition\nb. Subtraction\nc. Multiplication\nd. Division\n");

   scanf("%c", &choice);

         printf("Enter 2 integer numbers\n");

         scanf("%d %d", &a, &b);

         switch(choice)

   {

      case 'a': printf("%d + %d = %d\n", a, b, (a+b));

            break;

      case 'b': printf("%d - %d = %d\n", a, b, (a-b));

            break;

      case 'c': printf("%d x %d = %d\n", a, b, (a*b));

            break;

      case 'd':if( b != 0){

                        break;

            printf("%d / %d = %d\n", a, b, (a/b));

                                    }

            else{

               printf("Number can't be divided by 0\n");
```

```
                                    }
            break;
      default: printf("You entered wrong choice\n");
             break;
   }
         return 0;
}
```

Output:

Enter your choice

a. Addition

b. Subtraction

c. Multiplication

d. Division

c

Enter 2 integer numbers

99

75

99 x 75 = 7425

Practical 4

1. Write a C program to find the sum of individual digits of a positive integer.
Program:

```
#include<stdio.h>

#include<conio.h>


int main(){

	int n,sum=0;

	printf("Enter any Positive Integer =");

	scanf("%d",&n);

	while (n>0){

		sum=sum+n%10;

		n=n/10;

	}

	printf("Sum of Individual Digit of Integer is %d",sum);

	return 0;

}
```

Output:

```
Enter any Positive Integer =546832

Sum of Individual Digit of Integer is 28
```

2. A Fibonacci sequence is defined as follows: the first and second terms in the sequences 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

Program:

```
#include<stdio.h>

#include<conio.h>


int main(){

	int a=0,b=1,sum=0,length=0,counter;

	printf("Enter the length =");

	scanf("%d",&length);

	printf("Fibonacci sequence\n");

	printf("%d %d\n",a,b);

	for(counter=2; counter<length; counter++){

		sum=a+b;

		printf("%d\n",sum);

		a=b;

		b=sum;

	}

	return 0;

}
```

Output:

```
Enter the length =10

Fibonacci sequence

0 1

1 2 3 5 8 13 21 34
```

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

```
#include<stdio.h>
#include<conio.h>
int main()
{
  int n, i, j, count;
  printf("Prime no.series\n");
  printf("Enter any number\n");
  scanf("%d", &n);
  printf("The prime numbers between 1 to %d\n",n);
  for(i = 1; i <= n; i++)
  {
    count = 0;
    for(j = 1; j <=i; j++)
    if(i % j == 0)
    {
   count++;
    }
    if(count == 2)
    {
   printf("%d\t", i);
    }
  }
  return 0;
}
```

Output:

Prime no. series

Enter any number

50

The prime numbers between 1 to 50

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47

Practical 5

1. Write a C program to calculate the following Sum: Sum=1-x^2/2! +x^4/4!-x^6/6!+x^8/8!-x^10/10!

```
#include <stdio.h>

#include <math.h>


int main()

{

   int counter,f_coun;

   float sum=0,x,power,fact;

   printf("\tEQUATION SERIES : 1- X^2/2! + X^4/4! - X^6/6! + X^8/8! - X^10/10!");

   printf("\n\tENTER VALUE OF X : ");

   scanf("%f",&x);

   for(counter=0, power=0; power<=10; counter++,power=power+2)

   {

      fact=1;

      //Factorial of POWER value.

      for(f_coun=power; f_coun>=1; f_coun--)

         fact *= f_coun;

      //The main equation for sum of series is...

      sum=sum+(pow(-1,counter)*(pow(x,power)/fact));

   }

   printf("SUM : %f",sum);

   return 0;

}
```

Output:

EQUATION SERIES : 1- X^2/2! + X^4/4! - X^6/6! + X^8/8! - X^10/10!

ENTER VALUE OF X : 4

SUM : -0.685785

2. Write a C program to find the roots of a quadratic equation.

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

int main () {
   float a,b,c,r1,r2,d;

   printf ("Enter the values of a b c: ");
   scanf (" %f %f %f", &a, &b, &c);

   d= b*b - 4*a*c;

   if (d>0) {
      r1 = -b+sqrt (d) / (2*a);
      r2 = -b-sqrt (d) / (2*a);
      printf ("The real roots = %f %f", r1, r2);
   }
   else if (d==0) {
      r1 = -b/(2*a);
      r2 = -b/(2*a);
      printf ("Roots are equal =%f %f", r1, r2);
   }
   else
      printf("Roots are imaginary");

   return 0;
}
```

Output:

Enter the values of a b c: 2 10 4

The real roots = -7.938447 -12.061553

Practical 6

1. To find the factorial of a given integer (Without Recursion)

```
#include <stdio.h>
int main() {
    int n, i;
    unsigned long long fact = 1;
    printf("Enter an integer: ");
    scanf("%d", &n);

    // shows error if the user enters a negative integer
    if (n < 0)
        printf("Error! Factorial of a negative number doesn't exist.");
    else {
        for (i = 1; i <= n; ++i) {
            fact *= i;
        }
        printf("Factorial of %d = %llu", n, fact);
    }
    return 0;
}
```

Output:

```
Enter an integer: 9
Factorial of 9 = 362880
```

1. To find the factorial of a given integer (With Recursion)

```
#include<stdio.h>
long int multiplyNumbers(int n);
int main() {
   int n;
   printf("Enter a positive integer: ");
   scanf("%d",&n);
   printf("Factorial of %d = %ld", n, multiplyNumbers(n));
   return 0;
}

long int multiplyNumbers(int n) {
   if (n>=1)
      return n*multiplyNumbers(n-1);
   else
      return 1;
}
```

Output:

```
Enter a positive integer: 5
Factorial of 5 = 120
```

2. To find the GCD (greatest common divisor) of two given integers.

```
#include <stdio.h>
int main()
{
   int n1, n2, i, gcd;

   printf("Enter two integers: ");
   scanf("%d %d", &n1, &n2);

   for(i=1; i <= n1 && i <= n2; ++i)
   {
      // Checks if i is factor of both integers
      if(n1%i==0 && n2%i==0)
         gcd = i;
   }

   printf("G.C.D of %d and %d is %d", n1, n2, gcd);

   return 0;
}
```

Output:

```
Enter two integers: 12 18
G.C.D of 12 and 18 is 6
```

Practical 7

1. Write a C program to find the largest integer in a list of integers.

```
#include <stdio.h>

#include <conio.h>

int main()

{

  int a[25], i, large, small, n;

  printf("Enter the size of array(max 25)\n");

  scanf("%d", &n);

  printf("Enter any %d integer array elements\n",n);

  for(i = 0; i < n; i++)

  {

   scanf("%d", &a[i]);

  }

  large = a[0];

  small = a[0];

  for(i = 1; i < n ; i++)

  {

    if(a[i] > large)

    {

   large = a[i];

    }

    if(a[i] < small)

    {

   small = a[i];

    }
```

```
  }

  printf("The largest element from the given array is %d \nThe smallest element  from the given array is %d", large, small);

        return 0;

}
```

Output:

Enter the size of array(max 25)

5

Enter any 5 integer array elements

7 9 11 15 20

The largest element from the given array is 20

The smallest element from the given array is 7

2.1 Write a C program that uses functions to perform the following: Addition of Two Matrices

```
#include <stdio.h>
int main() {
  int r, c, a[100][100], b[100][100], sum[100][100], i, j;
  printf("Enter the number of rows (between 1 and 100): ");
  scanf("%d", &r);
  printf("Enter the number of columns (between 1 and 100): ");
  scanf("%d", &c);

  printf("\nEnter elements of 1st matrix:\n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("Enter element a%d%d: ", i + 1, j + 1);
      scanf("%d", &a[i][j]);
    }

  printf("Enter elements of 2nd matrix:\n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("Enter element b%d%d: ", i + 1, j + 1);
      scanf("%d", &b[i][j]);
    }

  // adding two matrices
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      sum[i][j] = a[i][j] + b[i][j];
    }

  // printing the result
  printf("\nSum of two matrices: \n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("%d   ", sum[i][j]);
      if (j == c - 1) {
        printf("\n\n");
      }
    }

  return 0;
}
```

Output:

Enter the number of rows (between 1 and 100): 3

Enter the number of columns (between 1 and 100): 3

Enter elements of 1st matrix:

Enter element a11: 1

Enter element a12: 2

Enter element a13: 3

Enter element a21: 4

Enter element a22: 5

Enter element a23: 6

Enter element a31: 7

Enter element a32: 8

Enter element a33: 9

Enter elements of 2nd matrix:

Enter element b11: 1

Enter element b12: 2

Enter element b13: 3

Enter element b21: 4

Enter element b22: 5

Enter element b23: 6

Enter element b31: 7

Enter element b32: 8

Enter element b33: 9

Sum of two matrices:

2 4 6

8 10 12

14 16 18

2.1 Write a C program that uses functions to perform the following: Multiplication of Two Matrices

```
#include<stdio.h>

int main() {

   int m, n, p, q, i, j, k;

   int a[10][10], b[10][10], res[10][10];


   printf("Enter the order of first matrix\n");

   scanf("%d%d", & m, & n);

   printf("Enter the order of second matrix\n");

   scanf("%d%d", & p, & q);


   if (n != p) {

      printf("Matrix is incompatible for multiplication\n");

   } else {

      printf("Enter the elements of Matrix-A:\n");

      for (i = 0; i < m; i++) {

         for (j = 0; j < n; j++) {

            scanf("%d", & a[i][j]);

         }

      }


      printf("Enter the elements of Matrix-B:\n");

      for (i = 0; i < p; i++) {

         for (j = 0; j < q; j++) {

            scanf("%d", & b[i][j]);

         }
```

```
    }

    for (i = 0; i < m; i++) {
        for (j = 0; j < q; j++) {
            res[i][j] = 0;
            for (k = 0; k < p; k++) {
                res[i][j] += a[i][k] * b[k][j];
            }
        }
    }

    printf("The product of the two matrices is:-\n");

    for (i = 0; i < m; i++) {
        for (j = 0; j < q; j++) {
            printf("%d\t", res[i][j]);
        }
        printf("\n");
    }
  }

  return 0;
}
```

Output:

Enter the order of first matrix

3 3

Enter the order of second matrix

3 3

Enter the elements of Matrix-A:

1 2 3

4 5 6

7 8 9

Enter the elements of Matrix-B:

1 2 3

4 5 6

7 8 9

The product of the two matrices is:-

30 36 42

66 81 96

102 126 150

Practical 9

1. 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.

```
#include <stdio.h>

#include <string.h>

int main() {

char s[30],t[20];

char*found;

puts("Enter the first string:");

gets(s);

puts("Enter the string to be searched:");

gets(t);

found=strstr(s,t);

if(found)

{

printf("Second string is found in the first string at %d position.\n",found-s);

}

else

{

printf("-1");

}

return 0;

}
```

Output :

Enter the first string:

hi how are you

Enter the string to be searched:

are

Second string is found in the first string at 7 position.

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

```
#include <stdio.h>
#include <string.h>
int main()
{
	char str[100];
	int i=0,l=0,line=0,f=1;
	puts("Enter any string");
	gets(str);
	for(i=0;str[i]!='\0';i++)
	{
		l=l+1;
	}
	printf("the number of character in the string are %d\n",l);
	for(i=0;i<=l-1;i++)
	{
		if(str[i]==' ')
		{
			f=f+1;
		}
	}
	printf("The number of words in the string are %d",f);
	return 0;
}
```

Output :

Enter any string

hello how are you

the number of character in the string are 17

The number of words in the string are 4

Practical 8

1. To insert a sub-string into a given main string from a given position.

```
#include <stdio.h>
#include <string.h>
int main()
{
char a[10];
char b[10];
char c[10];
int p=0,r=0,i=0;
int t=0;
int x,g,s,n,o;
puts("Enter First String:");
gets(a);
puts("Enter Second String:");
gets(b);
printf("Enter the position where the item has to be inserted: ");
scanf("%d",&p);
r = strlen(a);
n = strlen(b);
i=0;
// Copying the input string into another array
while(i <= r)
{
 c[i]=a[i];
 i++;
}
```

```
s = n+r;
o = p+n;
// Adding the sub-string
for(i=p;i<s;i++)
{
 x = c[i];
 if(t<n)
 {
  a[i] = b[t];
  t=t+1;
 }
 a[o]=x;
 o=o+1;
}
printf("%s", a);
return 0;
}
```

Output :

```
Enter First String:
hi
Enter Second String:
hello
Enter the position where the item has to be inserted: 1
hhelloi
```

2. To delete n Characters from a given position in a given string.

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

int main() {
    char str[] = "Hello, world!";
    int n = 5;
    int i, j;

    // Delete n characters from a given position in a given string
    for (i = 0; i < n; i++) {
        for (j = 0; str[j] != '\0'; j++) {
            str[j] = str[j + 1];
        }
    }
    str[j - 1] = '\0';

    // Print the updated string
    printf("The updated string is: %s\n", str);

    return 0;
}
```

Output :

The updated string is: , world!

3. Write a C program to determine if the given string is a palindrome or not.

```
#include <stdio.h>
#include <string.h>
int main()
{
   char str[100];
   int i, j, len;
   printf("Enter a string: ");
   scanf("%s", str);
   len = strlen(str);
   for (i = 0, j = len - 1; i < j; i++, j--)
   {
      if (str[i] != str[j])
      {
         printf("The string is not a palindrome.\n");
         return 0;
      }
   }
   printf("The string is a palindrome.\n");
   return 0;
}
```

Output :

```
Enter a string: nitin
The string is a palindrome.
```

Practical 10

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

```
#include <stdio.h>

int main() {
  int rows, coef = 1, space, i, j;
  printf("Enter the number of rows: ");
  scanf("%d", &rows);
  for (i = 0; i < rows; i++) {
    for (space = 1; space <= rows - i; space++)
      printf("  ");
    for (j = 0; j <= i; j++) {
      if (j == 0 || i == 0)
        coef = 1;
      else
        coef = coef * (i - j + 1) / j;
      printf("%4d", coef);
    }
    printf("\n");
  }
  return 0;
}
```

Output :

```
1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
```

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

```
#include <stdio.h>
int main() {
  int i, space, rows, k = 0;
  printf("Enter the number of rows: ");
  scanf("%d", &rows);
  for (i = 1; i <= rows; ++i, k = 0) {
    for (space = 1; space <= rows - i; ++space) {
      printf("  ");
    }
    while (k != 2 * i - 1) {
      printf("* ");
      ++k;
    }
    printf("\n");
  }
  return 0;
}
```

Output :

```
*
    * * *
   * * * * *
  * * * * * * *
* * * * * * * * *
```

Practical 11

1. Write a C program to read in two numbers, x and n, and then compute the sum of this geometric progression: 1+x+x^2+x^3+... x^n. 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 without computing the sum. Are any values of x also illegal? If so, test for them too.

```
#include <stdio.h>

#include <math.h>


int main()

{

   int sum=0, n, x, i;

   printf("Enter the limit\n");

   scanf("%d",&n);

   printf("Enter the value of x\n");

   scanf("%d",&x);


   if (x<0 || n<0)

   {

      printf("Illegal Value");

   }

   else

   {

      for (i=0; i<n; i++)

      {

         sum=sum+pow(x,i);
```

```
    }


  }
  printf("The sum of the geomatric progression is %d",sum);
  return 0;
}
```

OUTPUT :

Enter the limit

10

Enter the value of x

5

The sum of the geomatric progression is 2441406

Practical 12

1. 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.

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

int main()
{
   char a[50];
   int i, carry, l;
   printf("Enter the Binary Number :");
   scanf("%s",&a);
   l=strlen(a);

   for(i=0; i<l; i++)
   {
      if(a[i]=='0')
      {
         a[i]='1';
      }
      else
      {
         a[i]='0';
      }
   }

   printf("The 1's compliment of the binary number is %s\n",a);

   i=strlen(a)-1;

   while (i>=0)
   {
      if(a[i]=='0')
      {
         a[i]='1';
         carry=0;
         break;
      }
      else
      {
         a[i]='0';
         carry=1;
         i=i-1;
      }
   }
```

```
 printf("The 2's compliment of the binary number is %s");

   if(carry==1)
   {
      printf("1");
   }
   printf("%s",a);
   return 0;
}
```

OUTPUT :

Enter the Binary Number :111000

The 1's compliment of the binary number is 000111

The 2's compliment of the binary number is001000

2. Write a C program to convert a Roman numeral to its decimal Equivalent.

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

int main()
{
   char rom[30];
   int a[30], l, k, i, dec;
   printf("Enter the roman number\n");
   scanf("%s,&rom");
   l=strlen(rom);

   for(i=0; i<l; i++)
   {
      switch (rom[i])
      {
         case 'I' : a[i]=1;
         break;

         case 'V': a[i]=5;
         break;

         case 'X': a[i]=10;
         break;

         case 'L': a[i]=50;
```

```
        break;

        case 'C': a[i]=100;
        break;

        case 'D': dec=dec+500;
        break;

        case 'M': a[i]=1000;
        break;

        default: printf("Invalid Choice");
        break;
    }
}

k=a[l-1];
for(i=l-1; i>0; i--)
{
    if(a[i]>a[i-1])
    {
        k=k-a[i-1];
    }

    if(a[i]<=a[i-1])
    {
        k=k+a[i-1];
```

```
        }
    }
    printf("The integer value of  roman number %s is %d",rom,k);
    return 0;
}
```

OUTPUT :

Enter the roman number

XIV

Decimal equivalent is 14

Practical 13

1. Write a c program on Given an unsorted array arr[] of size N. Rotate the array to the left (counter-clockwise direction) by D steps, where D is a positive integer.

```
#include <stdio.h>

void leftRotate(int arr[], int n, int d)

{

   int temp[d];


   for (int i = 0; i < d; i++)

   {

      temp[i] = arr[i];

   }


   for (int i = d; i < n; i++)

   {

      arr[i - d] = arr[i];

   }


   for (int i = 0; i < d; i++)

   {

      arr[n - d + i] = temp[i];

   }

}


void printArray(int arr[], int size)

{

   for (int i = 0; i < size; i++)

   {
```

```
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int main()
{
    int n, d;

    printf("Enter the size of the array: ");
    scanf("%d", &n);

    int arr[n];

    printf("Enter the elements of the array:\n");

    for (int i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }

    printf("Enter the number of rotations (D): ");
    scanf("%d", &d);

    leftRotate(arr, n, d);

    printf("Rotated array: ");
```

```
    printArray(arr, n);


    return 0;
}
```

Output :

Enter the size of the array: 5

Enter the elements of the array:

1 2 3 4 5

Enter the number of rotations (D): 2

Rotated array: 3 4 5 1 2

2. Write a c Program on given two sorted arrays arr1 and arr2 of size N and M respectively and an element K. The task is to find the element that would be at the k'th position of the final sorted array.Explanation:
Input :
Array 1 - 1 4 2 3 5
Array 2 - 7 8 6
k = 5
Output : 5
Because The final sorted array would be -1, 2, 3, 4, 5, 6, 7, 8, The 5th element of this array is 6.

```
#include <stdio.h>

int kthElement(int arr1[], int arr2[], int N, int M, int K)

{

   int i = 0, j = 0, k = 0;


   while (i < N && j < M)

   {

      if (arr1[i] < arr2[j])

      {

         if (k == K - 1)

         {

            return arr1[i];

         }

         i++;

      }

      else

      {

         if (k == K - 1)

         {

            return arr2[j];

         }
```

```
            j++;
        }
        k++;
    }

    while (i < N)
    {
        if (k == K - 1)
        {
            return arr1[i];
        }
        i++;
        k++;
    }

    while (j < M)
    {
        if (k == K - 1)
        {
            return arr2[j];
        }
        j++;
        k++;
    }

    return -1;
}
```

```
int main()
{
    int N, M, K;

    printf("Enter the size of the first array (N): ");
    scanf("%d", &N);

    int arr1[N];

    printf("Enter the elements of the first array in sorted order:\n");
    for (int i = 0; i < N; i++)
    {
        scanf("%d", &arr1[i]);
    }

    printf("Enter the size of the second array (M): ");
    scanf("%d", &M);

    int arr2[M];

    printf("Enter the elements of the second array in sorted order:\n");
    for (int i = 0; i < M; i++)
    {
        scanf("%d", &arr2[i]);
    }
```

```
    printf("Enter the value of K: ");
    scanf("%d", &K);

    int result = kthElement(arr1, arr2, N, M, K);

    printf("The element at the %d'th position in the merged sorted array is: %d\n", K, result);

    return 0;
}
```

Output :

```
Enter the size of the first array (N): 5
Enter the elements of the first array in sorted order:
0 1 2 3 4
Enter the size of the second array (M): 5
Enter the elements of the second array in sorted order:
5
6
7
8
9
Enter the value of K: 6
The element at the 6'th position in the merged sorted array is: 5
```

1. Write a c program to take multiline string input and print individual string length .

```
#include <stdio.h>
#include <string.h>
int main()
{
   char input[100];
   printf("Enter multiline string (press Enter twice to end input):\n");

   while (fgets(input, sizeof(input), stdin) && input[0] != '\n')
   {
      input[strcspn(input, "\n")] = '\0';

      printf("Length of the string: %zu\n", strlen(input));
   }
   return 0;
}
```

Output :

```
Enter multiline string (press Enter twice to end input):
hi how are you
Length of the string: 15
my name is parth joshi
Length of the string: 22
i study btech in parul university
Length of the string: 33
```

2. Write a c program to reverse the individual word of a given string Explanation:input : Welcome To Bytexl output: emocleW oT lxetyB.

```
#include <stdio.h>

#include <string.h>

void reverseWord(char str[], int start, int end)

{

   while (start < end)

   {

      char temp = str[start];

      str[start] = str[end];

      str[end] = temp;

      start++;

      end--;

   }

}


void reverseWords(char str[])

{

   int start = 0;

   int end = 0;


   while (str[end] != '\0')

   {

      while (str[start] == ' ')

      {

         start++;

      }
```

```
        end = start;
        while (str[end] != ' ' && str[end] != '\0')
        {
            end++;
        }

        reverseWord(str, start, end - 1);

        start = end;
    }
}

int main()
{

    char input[100];

    printf("Enter a string: ");
    fgets(input, sizeof(input), stdin);

    input[strcspn(input, "\n")] = '\0';

    reverseWords(input);

    printf("Reversed string: %s\n", input);

    return 0;
```

```
}
```

Output :

```
Enter a string: hello my name is parth joshi

Reversed string: olleh ym eman si htrap ihsoj
```