



TABLE OF CONTENTS

Vision, Mission, Motto of Institute	I
Vision, Mission of Department	II
Program Educational Objectives (PEOs)	III
Program Specific Outcomes (PSOs)	IV
Program outcomes (POs)	V
Course outcomes of course (COs)	VI
CO-PO-PSO Mapping matrix	VII
LIST OF EXPERIMENT PRESCRIBED	VIII
Evaluation Rubrics	IX



I. INSTITUTE VISION-MISSION-MOTTO

I. Vision, Mission and Motto of the Institute:

Institute Vision:



ACHARYA INSTITUTE OF TECHNOLOGY
DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

(Affiliated to Visvesvaraya Technological University, Belagavi, Approved by AICTE, New Delhi and Accredited by NBA & NAAC)

Acharya Dr. Sarvepalli Radhakrishnan Road, Achitnagar Post, Soladevanahalli,
Bengaluru – 560107

Acharya Institute of Technology, committed to the cause of value-based education in all disciplines, envisions itself as a fountainhead of innovative human enterprise, with inspirational initiatives for Academic Excellence.

Institute Mission:

Acharya Institute of Technology strives to provide excellent academic ambiance to the students for achieving global standards of technical education, foster intellectual and personal development, meaningful research and ethical service to sustainable societal needs.

Motto of the Institute:

Nurturing Aspirations Supporting Growth

II. DEPARTMENT VISION & MISSION

II. Vision and Mission of the Department:

Department Vision:

Envisions to be recognized for quality education and research in the field of Computing, leading to creation of competent engineers, who are innovative and adaptable to the changing demands of industry and society.

Department Mission:



ACHARYA INSTITUTE OF TECHNOLOGY
DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

(Affiliated to Visvesvaraya Technological University, Belagavi, Approved by AICTE, New Delhi and Accredited by NBA & NAAC)

Acharya Dr. Sarvepalli Radhakrishnan Road, Achitnagar Post, Soladevanahalli,
Bengaluru – 560107

- Act as a nurturing ground for young computing aspirants to attain the excellence by imparting quality education and professional ethics.
- Collaborate with industries and provide exposure to latest tools/ technologies.
- Create an environment conducive for research and continuous learning

III. PROGRAM EDUCATIONAL OBJECTIVES [PEO's]

III. Program Educational Objectives:

PEO-1: Students shall, have a successful career in academia, R&D organizations, IT industry or pursue higher studies in specialized field of Information Science And Engineering and allied disciplines.

PEO-2: Students shall, be competent, creative and a valued professional in the chosen field

PEO-3: Students shall, engage in life-long learning, professional development and adapt to the working environment quickly



ACHARYA INSTITUTE OF TECHNOLOGY
DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

(Affiliated to Visvesvaraya Technological University, Belagavi, Approved by AICTE, New Delhi and Accredited by NBA & NAAC)

Acharya Dr. Sarvepalli Radhakrishnan Road, Achitnagar Post, Soladevanahalli,
Bengaluru – 560107

PEO-4: Students shall, become effective collaborators and exhibit high level of professionalism by leading or participating in addressing technical, business, environmental and societal challenges.

IV. PROGRAM OUTCOMES [PO'S]

IV. Program Outcomes:

ENGINEERING GRADUATES WILL BE ABLE TO:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

V. PROGRAM SPECIFIC OUTCOMES [PSO's]

V. Program Specific Outcomes:

- PSO-1:** Able to apply knowledge of information management and communication systems to provide secured solutions for real time engineering applications.
- PSO-2:** Apply best software engineering practices, modern tools and technologies to deliver quality products.



VI. COURSE OUTCOMES OF THE COURSE (COs)

VI. COURSE OUTCOMES OF THE COURSE:

CO-NUMBER	COURSE OUTCOME STATEMENT	RBT
CO1	Explain the basic concepts of computer system and C programming to learn the programming fundamentals.	BL2
CO2	Apply the c programming constructs to solve simple problems.	BL3
CO3	Write a C program using arrays and functions.	BL3
CO4	Apply the concepts of strings and pointer for memory management and data manipulation.	BL3
CO5	Write a C program using structures, unions and Enumerated data-types.	BL3
CO6	Implement programs using structures, strings, pointers and file I/O operations for a given problem.	BL3



VII. CO-PO MATRIX

VII. CO-PO MATRIX:

COs	Program Outcomes												Program Specific Outcomes		
	P01	P02	P03	P04	P05	P06	P07	P08	P09	P010	P011	P012	PS01	PS02	PS03
CO-1	3	-	-	-	2	-	-	-	2	-	-	2	-	-	1
CO-2	3	2	2	-	2	-	-	-	2	2	-	2	-	-	1
CO-3	3	2	2	-	2	-	-	-	2	-	-	2	-	-	1
CO-4	3	2	2	-	2	-	-	-	2	2	-	2	-	-	1
CO-5	3	2	2	-	2	-	-	-	2	-	-	2	-	-	1
CO-6	3	3	3		3				2	2		3			2

VIII. LIST OF PROGRAMS PRESCRIBED

VIII. LIST OF THE PROGRAMS:

SL	Name of Program
1	Simulation of a Simple Calculator
2	Compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.
3	An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit; for the next 100 units 90 paise per unit; beyond 300 units Rs 1 per unit. All users are charged a minimum of Rs. 100 as meter charge. If the total amount is more than Rs 400, then an additional surcharge of 15% of the total amount is charged. Write a program to read the name of the user, the number of units consumed, and print out the charges..
4	Write a C Program to display the following by reading the number of rows as input, <pre> 1 1 2 1 1 2 3 2 1 1 2 3 4 3 2 1 ----- nth row </pre>
5	Implement Binary Search on Integers
6	Implement Matrix multiplication and validate the rules of multiplication.
7	Compute $\sin(x)/\cos(x)$ using Taylor series approximation. Compare your result with the built-in library function. Print both the results with appropriate inferences.
8	Sort the given set of N numbers using Bubble sort..
9	Write functions to implement string operations such as compare, concatenate, and find string length. Use the parameter passing techniques
10	Implement structures to read, write and compute average- marks of the students, list the students scoring above and below the average marks for a class of N students.
11	Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of N real numbers
12	Write a C program to copy a text file to another, read both the input file name and target file name

IX. LAB IA EVALUATION RUBRICS

IX. Evaluation Rubrics for lab Programs (Max marks 25):

Sl No	Parameters	Mark (15)	4-5	2-3	0-1
1.	Writing Program/Logic (present week's/previous week's)	5	The student is able to write the program without errors or with minimal errors	The student is able to write the program with minor error	The student has written incomplete program with error or not attempted to write the program.
2.	Implementation in the target language with different inputs	5	Student is able to execute, debug, and test the program for all possible inputs/test cases.	Student is able to execute the program, but fails to debug, and test the program for all possible inputs/test cases.	Student has executed the program partially (fails to meet desired output) or does not execute the program.
3.	Record and Viva	5	Student Communicates the concepts and program effectively both orally and written.	Student is not able to effectively Communicates the concepts and program effectively both orally and written.	Student fails to submit the record on time or Does not answer any viva questions
	Parameters	Marks (50)	10(Write-up)	30(Execution)	10(Viva)
4.	Internal Assessment	10	Conducted for 50 Marks and Scaled down to 10M (20 Write-Up+20 Execution+10 Viva)		

PROGRAMS

1. Simulation of a Simple Calculator.

```
#include<stdio.h>
#include<stdlib.h>
void main()
{
    char op;
    float a, b, res;
    printf("Enter the operator \n"); scanf("%c",&op);
    printf("Enter the two operands \n"); scanf("%f%f",&a,&b);
    switch(op)
    {
        case '+' : res = a + b;
                    break;
        case '-' : res = a - b;
                    break;
        case '*' : res = a * b;
                    break;
        case '/' : if( b != 0)
                    {
                        res = a / b;
                    }
                    else
                    {
                        printf("Division not possible\n"); exit(0);
                    }
                    break;
        default : printf("invalid operator\n");
                    exit(0);
    }

    printf(" Result = %f \n", res);
}
```

Output:

1. Enter an Operator :

+

Enter two Operands:

20

23

Result =43.000000

2. Enter an Operator: -

Enter two Operands:

23

20

Result = 3.000000

Enter an Operator: *

Enter two Operands:

3

2

Result =6.000000

Enter an Operator: /

Enter two Operands:

6

2

Result = 3.0000

Enter an Operator :!

Enter two Operands:

6

2

Invalid Operator

Enter an Operator :

/

Enter two Operands:

2

0

Division not possible

2. Compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.

```
#include<stdio.h>
#include<math.h>
void main( )
{
    float a, b, c, disc, root1, root2, real, img;
    printf("Enter the coefficients a,b and c of the quadratic equation \n");
    scanf("%f%f%f",&a, &b, &c);
    disc = b * b - 4 * a * c;
    if(disc == 0)
    {
        printf(" Roots are real and Equal\n");
        root1 = root2 = - b / (2 * a);
        printf(" Root1 = %f and Root2 = %f\n", root1, root2);
    }
    else if(disc > 0)
    {
        printf(" Roots are real and Distinct\n");
        root1 = ( - b + sqrt(disc) ) / (2 * a);
        root2 = ( - b - sqrt(disc) ) / (2 * a);
        printf(" Root1 = %f and Root2 = %f\n", root1, root2);
    }
    else if(disc < 0)
    {
        printf(" Roots are Complex and Distinct\n");
        real = - b / (2 * a);
        img = sqrt(fabs(disc) ) / (2 * a);
        printf(" Root1 = %f + i %f \n", real, img);
        printf(" Root2 = %f - i %f \n", real, img);
    }
}
```

Output

1. Enter the coefficients a,b and c of the quadratic equation 1 1 1

Roots are Complex and Distinct

Root1=0.5000000+i0.866025

Root2=0.5000000-i0.866025

2. Enter the coefficients a,b and c of the quadratic equation 1 2 1

Roots are Real and Equal

Root1=-1.000000 and Root2=-1.000000

3. Enter the coefficients a,b and c of the quadratic equation 1 4 2

Roots are Real and distinct

Root1=-0.585786 and Root2=-3.414214

3. An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit; for the next 100 units 90 paise per unit; beyond 300 units Rs 1 per unit. All users are charged a minimum of Rs. 100 as meter charge. If the total amount is more than Rs 400, then an additional surcharge of 15% of total amount is charged. Write a program to read the name of the user, number of units consumed and print out the charges.

```
#include<stdio.h>
void main( )
{
    char name[20];
    int units;
    float charges, subcharge;
    printf("Enter the name of the customer:\n");
    scanf("%s", name);
    printf("Enter the total number of units consumed:\n");
    scanf("%d", &units);
    if(units >= 0 && units <=200)
    {
        charges = 100 + units*0.80;
    }
    else if(units > 200 && units <=300)
    {
        charges = 100 + 200 * 0.80 + (units - 200) * 0.90;
    }
    else if(units > 300)
    {
        charges = 100 + 200 * 0.80 + 100 * 0.90 + (units - 300) * 1.00;
    }

    if(charges > 400)
    {
        subcharge = charges * 0.15;
        charges = charges + subcharge;
    }
    printf("User %s need to pay %f rupees for electricity Bill", name, charges );
}
```

Output:

1. Enter the name of the customer:

Rani

Enter the total number of units consumed:

0

User Rani need to pay 100.000000 rupees for electricity Bill

2. Enter the name of the customer :

Ayush

Enter the total number of units consumed:

200

User Ayush need to pay 260.000000 rupees for electricity Bill

Enter the name of the customer:

Raju

Enter the total number of units consumed:

300

User Raju need to pay 350.000000 rupees for electricity Bill

4. Write a C Program to display the following by reading the number of rows input,

```
      1
    1 2 1
  1 2 3 2 1
1 2 3 4 3 2 1
- nth row
```

```
#include <stdio.h>
void main()
{
    int i, j, rows, k, m;
    printf (" Enter a number to define the rows: \n"); scanf ("%d", &rows);
    printf("\n");
    // outer loop define the total rows and i should be less than equal to rows
    for ( i = 1; i <= rows; i++)
    {
        printf(" "); // inner loop define j should be less
        for ( j = 1; j <= rows-i; j++)
        {
            printf (" ");
        }
        // use for loop where k is equal to 1
        for ( k = 1; k <= i; k++)
        {
            printf ("%d", k);
        }
        for (m = i-1; m >=1; m--)
        {
            printf("%d", m);
        }
        printf("\n");
    }
}
```

5. Implement Binary Search on Integers / Names.

```
#include<stdio.h>
#include<stdlib.h>
void main( )
{
    int n, a[20], key, low, high, mid ;
    printf("Enter the size of an array:\n");
    scanf("%d", &n);
    printf("Enter the elements to an array:\n");
    for( i = 0; i < n ; i++ )
    {
        scanf("%d", &a[i] );
    }
    printf("Enter the key element to be searched:\n");
    scanf("%d", &key);
    low = 0; high = n-1;
    while( low <= high )
    {
        mid = (low + high) / 2;
        if(key == a[mid])
        {
            printf("Element found at location: %d \n", mid + 1 );
            exit(0);
        }
        else if(key < a[mid])
            high = mid - 1;
        else
            low = mid +1;
    }
    printf("Key not found \n" );
}
```

Output:

1. Entre the size of array: 5

Enter the elements of an array: 10 20 30 40 50
Enter the key element to be searched: 30 Element
found at location 3

2. Entre the size of array: 5

Enter the elements of an array: 10 20 30 40 50
Enter the key element to be searched: 60
Key not found

6. Implement Matrix multiplication and validate the rules of multiplication.

```
#include<stdio.h>
void main( )
{
    int m, n, p, q, i, j, a[20][20], b[20][20], c[20][20];
    printf("Enter the number of rows and columns of matrix A:\n");
    scanf("%d %d", &m, &n);
    printf("Enter the number of rows and columns of matrix B:\n");
    scanf("%d %d", &p, &q);
    if(n!=p)
    {
        printf("Matrix multiplication is not possible\n"); exit(0);
    }
    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] );
        }
    }
    printf("The elements of matrix A are:\n");
    for( i = 0; i < m ; i ++ )
    {
        for( j = 0; j < n ; j ++ )
        {
            printf("%d \t", a[i][j] );
        }
        printf("\n");
    }

    printf("The elements of matrix B are:\n");
    for( i = 0; i < p ; i ++ )
    {
        for( j = 0; j < q ; j ++ )
        {
            printf("%d \t", b[i][j] );
        }
        printf("\n");
    }
    // To multiply the two matrices
```

```
for( i = 0; i < m ; i + + )
{
    for( j = 0; j < q ; j + + )
    {
        c[i][j] = 0;
        for( k = 0; k < n ; k + + )
        {
            c[i][j] = c[i][j] + a[i][k] * b[k][j];
        }
    }
}
printf("The resultant matrix is\n");
for( i = 0; i < m ; i + + )
{
    for( j = 0; j < q ; j + + )
    {
        printf("%d \t",c[i][j] );
    }
    printf("\n");
}
}
```

Output:

1. Enter the number of rows and columns of matrix A:

2 3

Enter the number of rows and columns of matrix B:

2 2

Matrix multiplication is not possible

2. Enter the number of rows and columns of matrix A:

2 2

Enter the number of rows and columns of matrix B:

2 2

Enter the elements of matrix A: 1 1 1 1

Enter the elements of matrix B: 1 1 1 1

The elements of matrix A are:

1 1

1 1

The elements of matrix B are:

1 1

1 1

The resultant matrix is:

2 2

2 2

7. Compute $\sin(x)/\cos(x)$ using Taylor series approximation. Compare your result with the built-in library function. Print both the results with appropriate inferences.

```
#include<stdio.h>
#define PI 3.142
void main( )
{
    int i, degree;
    float x, sum=0, term, num, deno;
    printf("Enter the value in degrees:\n");
    scanf("%d", &degree);
    x=degree * (PI / 180);
    num= x;
    deno=1; i = 2;
    do
    {
        term=num/deno;
        num=-num*x*x;
        sum=sum+term;
        deno = deno*i*(i+1));
        i = i + 2;
    }
    while(fabs(term)>=0.00001);
    printf("The sine of %d is: %f\n", degree, sum);
    printf("The sine function of %d is: %f\n", degree, sin(x));
}
```

Output:

```
1. Enter the values in degrees:90 The
   sine of 90 is 1.000000
The sin function of 90 is 1.000000
2.Enter the values in degrees:45
The sine of 45 is 0.706825
```

8. Sort the given set of N numbers using Bubble sort.

```
#include<stdio.h>
void main( )
{
    int n, i, j, temp, a[100];
    printf("Enter the value for n:\n");
    scanf("%d",&n);
    printf("Enter the array elements:\n"); for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    printf("The array elements before sorting are: \n"); for(i=0;i<n;i++)
    {
        printf("%d\n",a[i]);
    }
    for(i=0; i<n-1; i++)
    {
        for(j=0 ; j< n-i-1 ; j++)
        {
            if(a[j]>a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
    printf("The elements after sorting are: \n");
    for(i=0;i<n;i++)
    {
        printf("%d\n",a[i]);
    }
}
```

Output:

```
Enter the value for n: 5 Enter the
array elements:
10 90 40 60 20
The array elements before sorting are: 10 90
40 60 20
The elements after sorting are: 10 20
40 60 90
```

**9. Write functions to implement string operations such as compare, concatenate, string length.
Convince the parameter passing techniques.**

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void strcmp(char str1[30],char str2[30]); void
strcon(char str1[30],char str2[30]); void
strlenth(char str1[30],char str2[30]); void
main()
{
    int ch;
    char str1[30],str2[30];
    printf("Enter string1 and String2");
    scanf("%s%s",str1,str2);
    printf("Enter your choice:\n 1. For string compare\n 2:For string concatenation\n
    3:For string length:\n");
    scanf("%d",&ch);
    switch(ch)
    {
        case 1: strcmp(str1,str2); break;
        case 2: strcon(str1,str1);
                break;
        case 3: strlenth(str1,str2); break;
        default: printf("Invalid choice, try with valid input"); exit(0);
    }
}

void strcmp(char str1[], char str2[])
{
    int i;
    for(i=0;str1[i]!='\0';i++)
    {
        if(str1[i]!=str2[i])
        {
            printf("Strings are not equal\n");
            exit(0);
        }
    }
    printf("The string are equal\n");
}

void strcon(char str1[], char str2[])
{
    int i,len; len=strlen(str1);
    for(i=0;str2[i]!='\0';i++)
    {
        str1[len+i]=str2[i];
    }
}
```

```
str1[len+i]='\0';
printf("After concatenation String 2 to string 1 is: %s\n",str1);
}
void strlength(char str1[],char str2[])
{
int i,len1=0,len2=0;
for(i=0;str1[i]!='\0';i++)
{
len1++;
}
for(i=0;str2[i]!='\0';i++)
{
len2++;
}
printf("The length of string1 :%d\n length of string2: %d\n",len1,len2);
}
```

Output:

1. Enter string1 and String2

Acharya

Acharya

Enter your choice:

1: For string compare

2: For string concatenation 3: For

string length:

1

Strings are equal

2.Enter string1 and String2 Acharya

Acharya

Enter your choice:

1: For string compare

2: For string concatenation 3: For

string length:

1

Strings are equal

10. Implement structures to read, write and compute average- marks and the students scoring above and below the average marks for a class of N students.

```
#include<stdio.h>
struct student
{
char name[20]; int rollno;
int m1, m2, m3; int avg;
};
void main( )
{
struct student s[100];
int n, i, sum=0, class_avg =0;
printf("Enter the number of students:\n");
scanf("%d", &n);
// To read the details of 'n' students printf("Enter the
studentdetails\n"); for(i=0; i<n; i++)
{
printf("Enter the name: \n");
scanf("%s", s[i]. name);
printf("Enter the rollno:\n");
scanf("%d", &s[i]. rollno);
printf("Enter the marks in three tests:\n");
scanf("%d%d%d", &s[i].m1, &s[i].m2, &s[i].m3);
}
// To compute the average marks of each student
for(i=0; i<n; i++)
{
s[i].avg = ( s[i].m1 + s[i].m2 + s[i].m3) / 3;
}
// To compute the average marks of class
for(i=0; i<n; i++)
{
sum = sum + s[i].avg;
}
class_avg = sum / n;
printf("The average marks of class is :%d\n", class_avg);

// To print the names of students scoring above average

printf("Above average students:\n");
for(i=0; i<n; i++)
{
if( s[i].avg >= class_avg)
{
printf("%d\t", s[i].rollno);
printf("%s\n", s[i].name);
}
}
```

```
// To print the names of students scoring below average
printf("Below average students:\n"); for(i=0; i<n; i++)
{
    if( s[i].avg < class_avg)
    {
        printf("%d\t", s[i].rollno);
        printf("%s\n", s[i].name);
    }
}
```

Output:

```
1. Enter the number of students: 3 Enter
the student details
Enter the name:
Raju
Enter the roll no:
1
Enter the marks in three tests: 20 30
40
Enter the name:
Rani
Enter the roll no:
2
Enter the marks in three tests: 67 89
45
Enter the name:
Bhoomi
Enter the roll no:
3
Enter the marks in three tests:
90 45 66
The average marks of the class is :54 Above
average students:
0. Rani
0. Bhoomi
Below average students:
1. Raju
```

11. Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of 'n' real numbers.

$$\text{mean} = \frac{\sum_{i=0}^{n-1} a[i]}{n} \quad \text{variance} = \frac{\sum_{i=0}^{n-1} (a[i] - \text{mean})^2}{n} \quad \text{standard deviation} = \sqrt{\text{variance}}$$

```
#include<stdio.h>
void main( )
{
    int n, i,temp;
    float a[20], sum, mean, var, sd;
    float *p;
    printf("Enter the total number of elements:\n");
    scanf("%d", &n);
    printf("Enter the elements to an array:\n");
    for( i = 0; i < n ; i ++ )
    {
        scanf("%f", &a[i] );
    }
    p = a;    // p = &a[0];
    // To find sum and mean sum=0.0;
    for( i = 0; i < n ; i ++ )
    {
        sum = sum + *(p+i);
    }
    mean = sum/n;
    // To find variance temp=0.0;
    for( i = 0; i < n ; i ++ )
    {
        temp = temp + (*(p+i) - mean) * (*(p+i) - mean);
    }
    var = temp/n;
    // To find standard deviation
    sd = sqrt(var);
    printf("Sum = %f\n", sum);
    printf("Mean = %f\n", mean);
    printf("Variance = %f\n", var);
    printf("Standard deviation = %f\n", sd);
}
```

Output:

```
Enter the total number of elements:5 Enter
the elements to an array:
1 2 3 4 5
Sum=15.000000
Mean=3.000000
Variance=2.000000
Standard deviation=1.414214
```

12. Write a C program to copy a text file to another, read both the input file name and target file

```
#include <stdio.h>

#include <stdlib.h> // For exit() int
main()
{
    FILE *fptr1, *fptr2; char
    filename[100], c;

    printf("Enter the filename to open for reading \n");
    scanf("%s", filename);

    // Open one file for reading fptr1
    = fopen(filename, "r"); if (fptr1 ==
    NULL)
    {
        printf("Cannot open file %s \n", filename); exit(0);
    }

    printf("Enter the filename to open for writing \n");
    scanf("%s", filename);

    // Open another file for writing
    fptr2 = fopen(filename, "w"); if
    (fptr2 == NULL)
    {
        printf("Cannot open file %s \n", filename); exit(0);
    }

    // Read contents from file
    c = fgetc(fptr1); while (c
    != EOF)
    {
        fputc(c, fptr2); c =
        fgetc(fptr1);
    }
    printf("\nContents copied to %s", filename);
    fclose(fptr1);
    fclose(fptr2); return
    0;
}
```

OUTPUT:

```
Enter the filename to open for reading source.txt
Enter the filename to open for writing destination.txt
Contents copied to destination.txt
```




ACHARYA INSTITUTE OF TECHNOLOGY
DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

(Affiliated to Visvesvaraya Technological University, Belagavi, Approved by AICTE, New Delhi and Accredited by NBA & NAAC)

Acharya Dr. Sarvepalli Radhakrishnan Road, Achitnagar Post, Soladevanahalli,
Bengaluru – 560107
