ANNAMALAI UNIVERSITY

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.E. [____]

II – SEMESTER

ETSP207 : Computer Programming Laboratory

Name : ____

Reg. No. : ____

ANNAMALAI UNIVERSITY

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.E. [______________________________________]

II – SEMESTER

ETSP207 : Computer Programming Laboratory

Certified that this is a bona fide record of work done by Mr./Ms.____

Reg. No.____

of B.E. (______________________________________) in the ETSP207 : Computer Programming Laboratory during the Even semester of the academic year 2019 – 2020.

Staff in-charge | Internal Examiner

Annamalai nagar | External Examiner
Date: … /…. / 2020.

B.E.(All Programmes)

I Year II Semester

ETSP207 : Computer Programming Laboratory

Academic Year 2019-2020

Designed and Prepared by

Dr. R. Ragupathy

Dr. G. Ramachandran

Mr. A. Kanthimathinathan

CONTENTS

Ex. No. : 01 **Simple Computational Problems using Arithmetic Expressions**

Date :

Ex. No. 1(a) Arithmetic Operations

Aim:

To perform the arithmetic operations using a C program.

Algorithm:

1. Start
2. Get the data assigned to the respective variables.
3. Add the numbers and print the result
4. Multiply the numbers and print the result
5. Divide the numbers and print the result
6. Get the remainder and print the result
7. Pre Increment a and print the result
8. Post Increment b and print the result
9. Stop

Program:

```
#include <stdio.h>
#include <conio.h>
void main()
{
  int a = 21;  int b = 10;  int c ;
  clrscr();
  c = a + b;      printf(" Value of c = a+b  is %d\n", c );
  c = a - b;      printf(" Value of c = a-b is %d\n", c );
  c = a * b;      printf(" Value of c = a*b is %d\n", c );
  c = a / b;      printf(" Value of c = a/b is %d\n", c );
  c = a % b;      printf(" Value of c = a%%b is %d\n", c );
  c = ++a;        printf(" Value of c = ++a is %d\n", c );  /* pre-incrementation */
  c = b++;        printf(" Value of c = b++ is %d\n", c ); /* post-incrementation */
  getch();
}
```

Sample Input / Output:

```
Value of c = a+b is 31
Value of c = a-b is 11
Value of c = a*b is 210
Value of c = a/b is 2
Value of c = a%b is 1
Value of c = ++a is 22
Value of c = b++is 10
```

Result:

Thus, the C program for arithmetic operations has been successfully executed.

Ex. No. 1(b) Simple Interest Calculation

Aim:

To calculate simple interest using a C program

Algorithm:

```
1. Start
2. Get the values of P,N,R
3. Calculate SI = PNR/100
4. Print SI
5. Stop
```

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
 float p,n,r,si;
 clrscr();
 printf("\n Enter Principal, Rate of interest & Time :\n");
 scanf("%f%f%f",&p,&r,&n);
 si=(p*r*n)/100;
 printf("Simple Interest is :%8.2f",si);
 getch();
}
```

Sample Input / Output:

```
Enter Principal, Rate of interest & Time:
1000
25
5
Simple Interest is:     1250.00
```

Result:

Thus, the C program for finding simple interest has been successfully executed.

Ex. No. 1(c) Convert Temperature from Centigrade to Fahrenheit

Aim:

To convert temperature from Centigrade to Fahrenheit using a C program

Algorithm:

```
1. Start
2. Get the temperature in degree centigrade
3. Calculate Fahrenheit = 1.8*C+32
4. Print the temperature in Fahrenheit
5. Stop
```

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
  float c,f;
  clrscr();
  printf("\n Enter the Temperature in Centigrade: ");
  scanf("%f",&c);
  f=(1.8*c)+32;
  printf("\n Temperature in Fahrenheit is : %f",f);
  getch();
 }
```

Sample Input / Output:

```
Enter the Temperature in Centigrade: 78
Temperature in Fahrenheit is : 172.399994
```

Result:

Thus, the C program for converting temperature from Centigrade to Fahrenheit has been successfully executed.

Ex. No. 1(d) Calculate Percentage of Marks in Five Subjects

Aim:

To calculate percentage of marks obtained in five subjects using a C program.

Algorithm:

```
1. Start
2. Assign total as 500
3. Get the marks obtained for 5 subjects
4. Add the marks of 5 subjects and get the sum
5. Calculate percentage = (sum / total ) * 100
6. Print the percentage
7. Stop
```

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
  int s1,s2,s3,s4,s5,sum=0,total=500;
  float percent;
  clrscr();
  printf("Enter Marks of 5 Subjects: \n");
  scanf("%d%d%d%d%d",&s1,&s2,&s3,&s4,&s5);
  sum=s1+s2+s3+s4+s5;
  printf("\n Sum of 5 Subjects is %d",sum);
  percent=(float)sum/(float)total * 100;
  printf("\n Percentage is %f", percent);
  getch();
}
```

Sample Input / Output:

```
Enter Marks of 5 Subjects:
98
78
67
95
83
Sum of 5 subjects is 421
Percentage is 84.200000
```

Result:

Thus, the C program for calculating percentage of marks obtained in five subjects has been successfully executed.

Ex. No. 1(e) Calculate Area and Circumference of a Circle

Aim:

To find the area and circumference of a circle using C program

Algorithm:

```
1. Start
2. Read the value of radius of the circle
3. Calculate Area = $\pi*r*r$
4. Print the area of the circle
5. Calculate Circumference = $2*\pi*r$
6. Print the circumference of the circle
7. Stop
```

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
  float r;
  float pi=3.14,area,ci;
  clrscr();
  printf("Enter the Radius of a Circle:");
  scanf("%f",&r);
  area=pi*r*r;
  printf("\n Area of a Circle is %f ",area);
  ci=2*pi*r;
  printf("\n Circumference is %f",ci);
  getch();
}
```

Sample Input / Output:

```
Enter the Radius of a Circle:15
Area of a Circle is 706.500000
Circumference is 94.200005
```

Result:

Thus, the C program for calculating area and circumference of a circle has been successfully executed.

Ex. No. : 02 **Problems Involving *if-else* Statements**
Date :

Ex. No. 2(a) Compare Two Numbers using *if* Statement

Aim:

To compare two numbers using *if* statement in C language

Algorithm:

```
1. Start
2. Input the value of x and y
3. if (x>y) then print x is greater than y
4. if (x<y) then print x is less than y
5. if (x=y) then print x is equal to y
6. Stop
```

Program:

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

void main()
{
  int x, y;
  clrscr();
  printf("\n Enter the value of x:");
  scanf("%d", &x);
  printf("\n Enter the value of y:");
  scanf("%d", &y);
  if (x>y)
  {
   printf("\n x is greater than y");
  }
  if (x<y)
  {
   printf("\n x is less than y");
  }
  if (x==y)
  {
   printf("\n x is equal to y");
  }
  getch();
}
```

Sample Input / Output:

```
Enter the value of x: 20
Enter the value of y: 10
x is greater than y

Enter the value of x: 10
Enter the value of y: 20
x is less than y

Enter the value of x: 10
Enter the value of y: 10
x is equal to y
```

Result:

Thus, the C program for comparing two numbers has been successfully executed.

Ex. No. 2(b) Odd or Even using *if-else* Statement

Aim:

To find whether a number is odd or even using *if-else* statement in C language

Algorithm:

```
1. Start
2. Input number
3. Remainder = number%2
4. if(Remainder=0) then print “Number is even”
     otherwise print “Number is odd”
5. Stop
```

Program:

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

void main()
{
  int n;
  clrscr();
  printf("\n Enter a value to find odd or even:");
  scanf("%d",&n);
  if(n%2==0)
     printf("\n Number is even");
  else
     printf("\n Number is odd");
  getch();
}
```

Sample Input / Output:

```
Enter a value to find odd or even: 5
Number is odd

Enter a value to find odd or even: 8
Number is even
```

Result:

Thus, the C program for finding odd or even number has been successfully executed

Ex. No. 2(c) Calculate Electricity Bill using *else-if* ladder Statement

Aim:

To calculate the electricity bill using *else-if* ladder statement in C language.

Algorithm:

1. Start
2. Read unit consumed from user
3. If unit < = 50 then amount = unit * 0.50
4. If unit > 50 and unit <=100 then amount = 50*0.5 + (unit-50) * 0.75
5. If unit > 100 and unit <=250 then amount = 50*0.5 + 100*0.75 + (unit-150) * 1.20
6. If unit > 250 then amount = 50*0.5 + 100*0.75 + 100*1.20 + (unit-250) * 1.50
7. Calculate surcharge = amount * 0.20
8. Calculate net amount = amount + surcharge
9. Stop

Program:

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

void main()
{
  int cust_sno,unit;
  float amt,sur_charge,total_amt;
  char cust_name[25];
  clrscr();
  printf("\n Enter the customer service no :");
  scanf("%d",&cust_sno);
  printf("\n Enter the customer name :");
  scanf("%s",cust_name);
  printf("\n Enter the unit consumed by customer :");
 scanf("%d",&unit);
 if(unit <= 50)
  {
   amt = unit * 0.50;
  }
 else if(unit <= 150)
     {
      amt = (50*0.50)+((unit-50)*0.75);
     }
    else if(unit <= 250)
        {
        amt = (50*0.50)+(100*0.75)+((unit-150)*1.20);
        }
   else
        {
        amt = (50*0.50)+(100*0.75)+(100*1.20)+((unit-250)*1.50);
       }

sur_charge = amt*0.20;
total_amt = amt+sur_charge;
clrscr();
printf("\n\t\t\t Electricity Bill");
printf("\n Customer Service No:%d",cust_sno);
printf("\n Customer Name :%s",cust_name);
printf("\n Unit Consumed :%d",unit);
printf("\n Net Amount with Surcharge : Rs. %.2f",total_amt);
getch();
}
```

Sample Input / Output:

```
Enter the customer service no: 353
Enter the customer name: Ram
Enter the unit consumed by customer: 50

                                        Electricity Bill
Customer Service No: 353
Customer Name : Ram
Unit Consumed : 50
Net Amount with Surcharge   : Rs. 30.00

Enter the customer service no: 119
Enter the customer name: Radha
Enter the unit consumed by customer: 300

                                        Electricity Bill
Customer Service No : 119
Customer Name : Radha
Unit Consumed : 300
Net Amount with Surcharge   : Rs. 354.00
```

Result:

Thus, the C program to calculate electricity bill using *else-if* ladder statement has successfully completed.

Ex. No. : 03 **Problems involving *switch ... case* Statement**
Date :

Convert any number from 0 to 9 into word using *switch … case*

Aim:

To convert any number from 0 to 9 into word using *switch … case* statement in C language.

Algorithm:

1. Get the value for n from the user.
2. Check the entered value of n with the cases listed below. If the value of n matches with the case value, then execute corresponding statement only,
    a. n=0 : display “Entered number is Zero”
    b. n=1 : display “Entered number is One”
    c. n=2 : display “Entered number is Two”
    d. n=3 : display “Entered number is Three”
    e. n=4 : display “Entered number is Four”
    f. n=5 : display “Entered number is Five”
    g. n=6 : display “Entered number is Six”
    h. n=7 : display “Entered number is Seven”
    i. n=8 : display “Entered number is Eight”
    j. n=9 : display “Entered number is Nine”

    otherwise display “Enter the number only between 0 and 9”.
3. Stop

Program:

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

void main()
{
  int num;
  clrscr();
  printf("\n\nEnter a number from 0 to 9 :  ");
  scanf("%d",&num);
  switch(num)
  {
   case 0:
          printf("\nEntered number is Zero \n");
          break;
   case 1:
          printf("\nEntered number is One \n");
          break;
   case 2:
          printf("\nEntered number is Two \n");
          break;
   case 3:
          printf("\nEntered number is Three \n");
          break;
   case 4:
```

```
                printf("\nEntered number is Four \n");
                break;
        case 5:
                printf("\nEntered number is Five \n");
                break;
        case 6:
                printf("\nEntered number is Six \n");
                break;
        case 7:
                printf("\nEntered number is Seven \n");
                break;
        case 8:
                printf("\nEntered number is Eight \n");
                break;
        case 9:
                printf("\nEntered number is nine \n");
                break;
        default:
                printf("\n Enter the number only from 0 to 9 \n");
                break;
        }
      getch();
    }
```

Sample Input / Output:

```
Enter a number from 0 to 9 : 0
Entered number is Zero

Enter a number from 0 to 9 : 1
Entered number is One

Enter a number from 0 to 9 : 9
Entered number is Nine

Enter a number from 0 to 9 :10
Enter the number only from 0 to 9
```

Result:

Thus, the C program to convert the given number into word using *switch … case* statement is executed successfully.

Ex. No. : 04 **Iterative problems using Loops (*while, do ... while, and for*)**
Date :

Ex. No. 4(a) Sum of Natural Numbers using *for* Statement

Aim:

To compute the sum of first n natural numbers using *for* looping statement in C language

Algorithm:

```
1. Start
2. Get the value of n.
3. Assign the initial value 0 to sum.
4. Assign the first natural number 1 to the variable i.
5. Repeat the following steps for n number of times
    a. Accumulate the present value of i along with the present value of sum in
       the variable sum. [i.e. sum = sum + i.]
    b. Increment the value of i by one. [i.e. i  = i + 1]
6. Display the result available in the variable sum.
7. Stop
```

Program:

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

void main()
{
 int n, i, sum = 0;
 clrscr();

 printf("Enter a Positive Integer Number: ");
 scanf("%d", &n);
 printf("\nThe Sum of the Series ");

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

 printf("\nSum = %d", sum);

 getch();
}
```

Sample Input / Output:

```
Enter a Positive Integer Number: 5

The Sum of the Series 1 + 2 + 3 + 4 +5 +
Sum = 15

Enter a Positive Integer Number: 1

The Sum of the Series 1 +
Sum = 1
```

Result:

Thus, the C program to find the sum of first n natural number using *for* looping statement is executed successfully.

Ex. No. 4(b) Sum of Even Numbers using *while* Looping Statement

Aim:

To compute the sum of first n even numbers using *while* looping statement in C language.

Algorithm:

```
1. Start
2. Get the value of n.
3. Assign the initial value 0 to sum.
4. Assign the first Even number 2 to the variable even.
5. Repeat the following steps for n number of times
    a. Accumulate the present value of even along with the present value of sum
       in the variable sum. [i.e. sum=sum +even.]
    b. Increment the value of even by two. [i.e. even = even + 2]
6. Display the result available in the variable sum.
7. Stop
```

Program:

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

void main()
{
 int n, i, sum = 0,even=2;
 clrscr();

 printf("Enter a Positive Integer Number: ");
 scanf("%d", &n);
 i = 0;
 printf("\nThe Even Number(s) are ");

 while ( i<n )
 {
  printf("%d ",even);
  sum += even;
  even +=2;
  i++;
 }

 printf("\nSum of above Even Number Series = %d", sum);

 getch();
}
```

Sample Input / Output:

```
Enter a Positive Integer Number: 5

The Even Number(s) are 2 4 6 8 10
Sum of above Even Number Series = 30

Enter a Positive Integer Number: 1

The Even Number(s) are 2
Sum of above Even Number Series = 2

Enter a Positive Integer Number: 0

The Even Number(s) are
Sum of above Even Number Series = 0
```

Result:

Thus, the C program to find the sum of first n even numbers using *while* looping statement is executed successfully.

Ex. No. 4(c) Sum of Odd Numbers using *do* ... *while* Looping Statement

Aim:

To compute the sum of first n odd numbers using *do … while* looping statement in C language.

Algorithm:

1. Start
2. Get the value of n.
3. Assign the initial value 0 to sum.
4. Assign the first Odd number 1 to the variable odd.
5. Repeat the following steps for n number of times
   a. Accumulate the present value of odd along with the present value of sum in the variable sum. [i.e. sum=sum + odd.]
   b. Increment the value of even by two. [i.e. odd = odd + 2]
6. Display the result available in the variable sum.
7. Stop

Program:

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

void main()
{
 int n, i=1, sum = 0,odd=1;
 clrscr();

 printf("Enter a Positive Integer Number : ");
 scanf("%d", &n);
 printf("\nThe Odd Numbers are ");

 if(n>=1)
 {

  do
  {
   printf("%d ",odd);
   sum += odd;
   i++;
   odd += 2;
  }while ( i<=n );

}
 printf("\nSum of above Odd Numbers Series = %d", sum);

 getch();
}
```

Sample Input / Output:

```
Enter a Positive Integer Number : 5

The Odd Number(s) are 1 3 5 7 9
Sum of above Odd Number Series = 25

Enter a Positive Integer Number : 1

The Odd Number(s) are 1
Sum of above Odd Number Series = 1

Enter a Positive Integer Number : 0

The Odd Number(s) are
Sum of above Odd Number Series = 0
```

Result:

Thus, the C program to find the sum of first n odd numbers by using *do… while* looping statement is executed successfully.

Ex. No. : 05 **Problems involving One Dimensional Array Manipulations**
Date :

Ex. No. 5(a) Searching an Element in the List of Numbers

Aim:

To search an element in the list using one dimensional array in C language

Algorithm:

```
1. Start
2. Get the number of elements in the list in variable n.
3. Assign the list of n numbers in the array.
4. Get the element e to be searched in the list.
5. Compare the element e with each element present in the array.
      a. If it is present then display “Element Present” along with its location value and
         go to step 7.
6. display “Element not found in the list”
7. Stop
```

Program:

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

void main()
{
 int a[30], e, num, i;
 clrscr();

 printf("\nEnter Number of Elements in the List : ");
 scanf("%d", &num);
 printf("\nEnter the List of Elements : \n");

 for (i = 0; i<num; i++)
 {
  scanf("%d", &a[i]);
 }

//Read the element to be searched
 printf("\nEnter the Value of Elements to be Searched : ");
 scanf("%d", &e);

//Search starts from the zeroth location i = 0;
 i=0;
 while (i<num)
 {
//Compare the Search Element with the values present at each location...
  if(e == a[i])
  {
   printf("\nThe Element is Present at the Location %d ",i+1);
   getch();
   exit(0);
  }
  i++;
 }

 printf("Element Not found in the List");
 getch();
}
```

Sample Input / Output:

```
Enter Number of Elements in the List : 7

Enter the List of Elements :
54 23 45 19 1 49 46

Enter the Value of Element to be Searched : 45

The Element is Present at the Location 3

Enter Number of Elements in the List : 8

Enter the List of Elements :
54 23 45 19 1 49 46 22

Enter the Value of Element to be Searched : 145

Element Not found in the List
```

Result:

Thus, the C program to search an element in the list using one dimensional array is executed successfully.

Ex. No. 5(b) Sorting the List of Numbers using Bubble Sort

Aim:

To sort the list of numbers in ascending order using Bubble sort algorithm.

Algorithm:

1. Start
2. Get the number of elements in the list in variable n.
3. Get the values of n elements in the array.
4. If the current element is greater than the next element of the array then interchange its positions.
5. Repeat the Step 4 for n-1 number of times.
6. Repeat Step 5 for (n-1-number of times step 5 executed) number of times.
7. Display the resultant array which is in ascending order.
8. Stop

Program:

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

void main()
{
  int array[100], n, i, j, temp;
  clrscr();
  printf("\nEnter the Number of Elements in the list to be Sorted:");
  scanf("%d", &n);

  printf("\nEnter %d Integers\n\n", n);

  for (i = 0; i < n; i++)
    scanf("%d", &array[i]);

  for (i = 0 ; i < n - 1; i++)
  {
    for (j = 0 ; j < n - i - 1; j++)
    {
      if (array[j] > array[j+1])
      {
            temp      = array[j];
            array[j]  = array[j+1];
            array[j+1] = temp;
      }
    }
  }

  printf("\nSorted list of Numbers in ascending order … \n");
  for (i = 0; i < n; i++)
     printf(" %d ", array[i]);
  getch();
}
```

Sample Input / Output:

```
Enter the Number of Elements in the list to be Sorted : 5

Enter 5 Integers

14 33 27 35 10

Sorted list of Numbers in ascending order …
 10 14 27 33 35
```

Result:

Thus, the C program to arrange the list of numbers in ascending order using Bubble sort algorithm is executed successfully.

Ex. No. 5(c) Sorting the List of Numbers using Selection Sort

Aim:

To sort the list of numbers in ascending order using Selection sort algorithm.

Algorithm:

1. Start
2. Get the number of elements in the list in variable n.
3. Get the values of n elements in the array.
4. Repeat the Step 5 for all i such that 0 <i < n.
5. Select the ith smallest element in the unsorted list and interchange it with the ith location.
6. Display the resultant array which is in ascending order.
7. Stop

Program:

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

void main()
{
  int array[100], n, i, j, position, t;
  clrscr();

  printf("Enter the Number of Elements in the List to be sorted : ");
  scanf("%d", &n);

  printf("nEnter %d Integers\n", n);

  for (i = 0; i < n; i++)
    scanf("%d", &array[i]);

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

    for (j = i + 1; j < n; j++)
    {
      if (array[position] > array[j])
            position = j;
    }
    if (position != i)
    {
      t = array[i];
      array[i] = array[position];
      array[position] = t;
    }
  }

  printf("\nSorted list of Numbers in ascending order: \n");
  for (c = 0; c < n; c++)
    printf("%d\n", array[c]);

  getch();
}
```

Sample Input / Output:

```
Enter the Number of Elements in the list to be Sorted : 7

Enter 7 Integers
45 23 19 8 -1 -3 -17

Sorted list of Numbers in ascending order :
-17 -3 -1 8 19 23 45
```

Result:

Thus, the C program to arrange the list of numbers in ascending order using Selection sort algorithm is executed successfully.

Ex. No. : 06 **Matrix Problems and String Operations**
Date :

Ex. No. 6(a) Matrix Addition

Aim:

To perform addition of two matrixes using two dimensional arrays in C language

Algorithm:

1. Start
2. Get the dimensions of the matrix A and B in row and col.
3. Get the elements of the matrices A
4. Get the elements of the matrices B
5. Perform addition of each value present in Matrix A at the location (i, j) with corresponding locations of Matrix B and store it in Matrix C.
   That is C[i, j] = A[i, j] + B[i, j] for all i, j such that 0 ≤ i < row and 0 ≤ j < col.
6. Display the resultant matrix C in the matrix format.
7. Stop

Program:

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

void main()
{
 int row, col, a[10][10], b[10][10], c[10][10], i, j;
 clrscr();

 printf("Enter the Dimension of the Matrix\n");
 printf("\nEnter the number of rows (between 1 and10): ");
 scanf("%d",&row);
 printf("\nEnter the number of columns (between 1and 10): ");
 scanf("%d",&col);

 printf("\nEnter the Elements of 1st Matrix in Matrix Format : \n");
 for(i=0; i<row; i++)
 {
  for(j=0; j<col; j++)
  {
   scanf("%d",&a[i][j]);
  }
 }

 printf("\nEnter the Elements of 2nd matrix in Matrix Format : \n");
 for(i=0; i<row; i++)
 {
  for(j=0; j<col; j++)
  {
   scanf("%d", &b[i][j]);
  }
 }
for(i=0;i<row;i++)
 {
  for(j=0;j<col;j++)
  {
   c[i][j]=a[i][j]+b[i][j];
  }
 }

printf("\n Addition of two Matrices is: \n\n");
 for(i=0;i<row;i++)
 {
  for(j=0;j<col;j++)
  {
   printf("%d ",c[i][j]);
  }
  printf("\n");
 }

 getch();
}
```

Sample Input / Output:

```
Enter the Dimension of the Matrix

Enter the number of rows (between 1 and 10) : 3

Enter the number of columns  (between 1 and 10) : 3

Enter the Elements of the 1st Matrix in Matrix Format :
1 2 3
4 5 6
7 8 1

Enter the Elements of the 1st Matrix in Matrix Format :
7 6 3
3 2 1
2 0 5

Addition of two Matrices is :

8 8 6
7 7 7
9 8 6
```

Result:

Thus, the C program to perform matrix addition using two dimensional array is executed successfully.

Ex. No. 6(b) **Matrix Multiplication**

Aim:

To perform multiplication of two matrixes using two dimensional array in C language

Algorithm:

1. Start
2. Get the dimensions of the matrices A in r1 and c1.
3. Get the dimensions of the matrices B in r2 and c2.
4. If the number of columns c1 of Matrix A is equal to number of rows r2 of Matrix B is equal then proceed to the next step 5otherwise stop the process after displaying "Multiplication Not Possible".
5. Get the elements of the matrices A
6. Get the elements of the matrices B
7. Repeat Step 8 for all i such that $0 \leq i < r1$
8. Repeat Step 9 for all j such that $0 \leq j < c2$
9. Repeat Step 10 for all k such that $0 \leq k < c1$
10. C[i][j]=C[i][j]+A[i][k]*B[k][j]
11. Display the resultant matrix C in the matrix format.
12. Stop

Program:

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

void main()
{
 int a[10][10], b[10][10], c[10][10], r1, c1, r2, c2, i, j, k;
 clrscr();

 printf("\nEnter Number of Rows and Column present in the First Matrix          : ");
 scanf("%d %d", &r1, &c1);
 printf("\nEnter Number of Rows and Column present in the Second Matrix         : ");
 scanf("%d %d",&r2, &c2);

if(c1!=r2)
 {
  printf("\n\nError! \n\nNumber of Columns of first matrix not equal to Number of rows of
second Matrix!!\n\n");
  printf("Matrix Multiplication cannot be Performed");
  getch();
  exit(0);
 }

printf("\nEnter the Elements of First Matrix in Matrix Format : \n");
 for(i=0; i<r1; i++)
  for(j=0; j<c1; j++)
   scanf("%d", &a[i][j]);

printf("\nEnter the Elements of Second Matrix in Matrix Format : \n");
```

```
 for(i=0; i<r2; i++)
  for(j=0; j<c2; j++)
   scanf("%d",&b[i][j]);


 for(i=0; i<r1; i++)
  for(j=0; j<c2; j++)
   c[i][j] = 0;

for(i=0; i<r1; i++)
 {
  for(j=0; j<c2; j++)
  {
   for(k=0; k<c1;k++)
   {
    c[i][j]=c[i][j]+a[i][k]*b[k][j];
   }
  }
 }

printf("\n Product of the two Matrices \n");
 for(i=0; i<r1; i++)
 {
  for(j=0; j<c2; j++)
  {
   printf("%d ", result[i][j]);
  }
  printf("\n");
 }

getch();
}
```

Sample Input / Output:

```
Enter the Number of Rows and Columns present in the First Matrix : 3 2

Enter the Number of Rows and Columns present in the Second Matrix : 2 3

Enter the Elements of the First Matrix in Matrix Format :
1 2
1 1
2 2

Enter the Elements of the Second Matrix in Matrix Format :
1 2 2
1 1 1

Product of the two Matrices
3 4 4
2 3 3
4 6 6
```

```
Enter the Number of Rows and Columns present in the First Matrix : 3 2

Enter the Number of Rows and Columns present in the Second Matrix       : 2 3

Error!

Number of Columns of first matrix not equal to Number of rows of second Matrix!!

Matrix Multiplication cannot be Performed.
```

Result:

Thus, the C program to perform matrix multiplication using two dimensional array is executed successfully.

.

Ex. No. 6(c) String Length using and without using Library Functions

Aim:

To find length of the given string using and without using library function in C language

Algorithm:

1. Start
2. Get the string in the variable str.
3. Call strlen( ) to find the length of the string and display it.
4. Start counting from first character to last character.
5. Display the count as length.
6. Stop

Program:

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

void main()
{
  char str[100];
  int i;
  int length = 0;

  clrscr();
  fflush(stdin);
  printf("\nSTRING OPERATIONS \n");
  printf("\nString Length \n");
  printf("\nEnter a string: ");
  scanf("%s",str);
  printf("\n\nLength of the string (using Library Function): %d", strlen(str));

  for(i = 0; str[i] != '\0'; ++i);
  length=i;
  printf("\n\nLength of the string (Without using Library function) : %d", length);
  getch();
 }
```

Sample Input / Output:

```
STRING OPERATIONS

String Length

Enter a string: Kanthimathinathan

Length of the string (using Library Function): 17
Length of the string (Without using Library function) : 17
```

Result:

Thus, the C program to find the length of the given string using and without using library function is executed successfully.

Ex. No. 6(d) String copy using and without using Library Functions

Aim:

To copy a string from one variable to another variable using and without using library function in C language

Algorithm:

1. Start
2. Get the string to be copied in the variable str1.
3. Call strcpy( ) to copy a string to another string.
4. Display original string and copied string.
5. Start copying from first character to last character to another string.
6. Display original string and copied string.
7. Stop

Program:

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

void main()
{
 char str1[100],str2[100],str3[100];
 int i;

 clrscr();
 fflush(stdin);
 printf("\nSTRING OPERATIONS \n");
 printf("\nString Copy \n");
 printf("\nEnter a string: ");
 scanf("%s",str1);

 strcpy(str2,str1);
 printf("\n\n\nAfter Copying (Using Library function) :");
 printf("\n\n Original String :%s",str1);
 printf("\n Copied String   :%s",str2);

 for(i = 0; str1[i] != '\0'; i++)
 {
  str3[i] = str1[i];
 }
 str3[i] = '\0';
 printf("\n\nAfter Copying (Without using Library Function): ");
 printf("\n\n Original String :%s",str1);
 printf("\n Copied String   :%s",str3);

 getch();
 }
```

Sample Input / Output:

```
STRING OPERATIONS

String Copy

Enter a string: Annamalai

After Copying (Using Library function) :

 Original String : Annamalai
 Copied String   : Annamalai

After Copying (Without using Library Function):

 Original String : Annamalai
 Copied String   : Annamalai
```

Result:

Thus, the C program to copy a string in one variable to another variable using and without using library function is executed successfully.

Ex. No. 6(e) String Concatenation using and without using Library Functions

Aim:

To concatenate two given strings using and without using library function in C language

Algorithm:

1. Start
2. Get first string in the variable str1.
3. Get second string in the variable str2.
4. Call strcat( ) to concatenate two strings.
5. Display concatenated string.
6. Get first string in the variable str1.
7. Get second string in the variable str2.
8. Start copying from first character to last character of second string into first string after the last character of it.
9. Display concatenated string.
10. Stop

Program:

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

void main()
{
 char str1[100],str2[100];
 int i,j;

 clrscr();
 fflush(stdin);
 printf("\n\t\tSTRING OPERATIONS \n");
 printf("\n\t\tString Concatenation \n");

 printf("\nUsing Library Function\n");
 printf("\nEnter first string: ");
 scanf("%s", str1);
 printf("\nEnter second string: ");
 scanf("%s", str2);
 strcat(str1,str2);
 printf("\nThe Concatenated string is  %s\n",str1);

 printf("\n\nWithout using Library Function\n");
 printf("\nEnter first string: ");
 scanf("%s", str1);
 printf("\nEnter second string: ");
 scanf("%s", str2);
 for(i = 0; str1[i] != '\0'; i++);
 for(j = 0; str2[j] != '\0'; j++, i++)
 {
  str1[i] = str2[j];
 }
 str1[i] = '\0';
 printf("\nThe Concatenated string is  %s\n",str1);

 getch();
 }
```

Sample Input / Output:

```
                STRING OPERATIONS

                String Concatenation

Using Library Function

Enter first string: Ramachandran

Enter second string: Gopalakrishnan

The Concatenated string is RamachandranGopalakrishnan


Without using Library Function

Enter first string: Kanthimathinathan

Enter second string:  Arunachalam

The Concatenated string is KanthimathinathanArunachalam
```

Result:

Thus, the C program to concatenate the two given strings using and without using library function is executed successfully.

Ex. No. 6(f) String Reverse using and without using Library Functions

Aim:

To reverse a given string using and without using library function in C language.

Algorithm:

```
1. Start
2. Get the string to be copied in the variable str.
3. Call strrev( ) to reverse a string.
4. Display reversed string.
5. Start copying from last character to first character to another string.
6. Display reversed string.
7. Stop
```

Program:

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

void main()
{
 char str[100],str2[100];
 int i, j, length;

 clrscr();
 fflush(stdin);
 printf("\n\t\tSTRING OPERATIONS \n");
 printf("\n\t\tString Reverse \n");
 printf("\nUsing Library Function\n");

 printf("\nEnter a string: ");
 scanf("%s",str);
 strrev(str);
 printf("\n Reversed string is: %s", str);

 printf("\n\n\nWithout using Library Function\n");
 printf("\nEnter a string: ");
 scanf("%s",str);
 length=strlen(str);
 i=0;j=length-1;
 while (i < length)
 {
   str2[i]=str[j];
   i++;
   j--;
  }
  str2[i]=’\0’;
 printf("\n Reversed string is: %s", str2);

 getch();
}
```

Sample Input / Output:

```
                STRING OPERATIONS

                String Reverse

Using Library Function

Enter a string: Apple

 Reversed string is: elppA


Without using Library Function

Enter a string: Orange

 Reversed string is: egnarO
```

Result:

Thus, the C program to reverse the given string using and without using library function is executed successfully.

Ex. No. 6(g) String Comparison using and without using Library Functions

Aim:

To compare two given strings using and without using library function in C language

Algorithm:

1. Start
2. Get first string in the variable str1.
3. Get second string in the variable str2.
4. Call strcmp( ) to compare two strings.
5. If result = 0 then display the result as both are equal else display the result as both are not equal.
6. Start comparing from first character to last character of first string with second string.
7. If all the characters are same then display the result as both are equal else display the result as both are not equal.
8. Stop

Program:

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

void main()
{
 char str1[100],str2[100];
 int i;

 clrscr();
 fflush(stdin);

 printf("\nSTRING OPERATIONS \n");
 printf("\nString Compare \n");

 printf("Enter the First String: ");
 scanf("%s", str1);
 printf("Enter Second String: ");
 scanf("%s", str2);

 printf("\n\nAfter Comparing two Strings (Using Library function) :");
 if(strcmp(str1,str2)==0)
 {
  printf("\nBoth the Strings are Same.\n");
 }
 else
 {
  printf("\nFirst String is not equal to Second String.\n");
 }

 printf("\n\nAfter Comparing two Strings (Without using Library function) :\n");
 i = 0;
 while (str1[i] == str2[i] && str1[i] != '\0')
  i++;
 if ((str1[i] > str2[i])||(str1[i] < str2[i]))
  printf("%s is not equal to %s", str1, str2);
 else
  printf("%s is equal to %s", str1, str2);

 getch();
 }
```

Sample Input / Output:

```
STRING OPERATIONS

String Compare
Enter the First String: Annamalai
Enter Second String: University


After Comparing two Strings (Using Library function) :
First String is not equal to Second String.


After Comparing two Strings (Without using Library function) :
Annamalai is not equal to University


STRING OPERATIONS

String Compare
Enter the First String: Annamalai
Enter Second String:  Annamalai

After Comparing two Strings (Using Library function) :
Both the Strings are Same.

After Comparing two Strings (Without using Library function) :
Annamalai is equal to Annamalai
```

Result:

Thus, the C program to compare two given strings using and without using library function is executed successfully.

Ex. No. : 07 **Problems involving Simple Functions**
Date :

Ex. No. 7(a) Swapping of Numbers using Function Call by Value

Aim:

To swap the contents available in two variables using function call by value.

Algorithm:

1. Start
2. Assign two different values to two integer variables let a =10 and b = 20.
3. Display the present value of a and b.
4. Call the Function swap by passing the values of a and b
5. Display the present value of a and b.
6. Stop

Function swap(v1, v2)

1. Display the present value of v1 and v2.
2. Using temp variable interchange the contents of the variable v1 and v2.
3. Display the present value of v1 and v2.
4. Return to the called place.

Program:

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

void swap(int , int);

void main()
{
  int a = 10;
  int b = 20;
  clrscr();

  printf("From Main \n Before Swapping the Values \n a = %d, b = %d\n",a,b);
  swap(a,b);

  printf("\n From Main \n After Swapping Values at Function Swap a = %d, b = %d\n",a,b);
  getch();
}

void swap (int v1, int v2)
{
  int temp;
  printf("\nInside Function Swap");
  temp = v1;
  v1=v2;
  v2=temp;
  printf("\nAfter Swapping Values inside Function Swap \n v1 = %d, v2 = %d\n",v1,v2);
  printf("Exiting from Function Swap\n");
}
```

Sample Input / Output:

```
From Main
Before Swapping the Values
a = 10, b = 20

Inside Function Swap
After Swapping Values inside Function Swap
v1 = 20, v2 = 10
Exiting from Function Swap

From Main
After Swapping Values at Function Swap
a= 10, b = 20
```

Result:

Thus, the C program to swap the contents available in two variables using function call by value is executed successfully.

Ex. No. 7(b) Swapping of Numbers using Function Call by Reference

Aim:

To swap the contents available in two variables using function call by reference.

Algorithm:

1. Start
2. Assign two different values to two integer variables a =10 and b = 20.
3. Display the present value of a and b.
4. Call the Function swap by passing the address of the variable a and b
5. Display the present value of a and b.
6. Stop

Function swap(ptr1, ptr2)

1. Display the present value of ptr1 and ptr2.
2. Using temp variable interchange the contents of the pointer variable ptr1 and ptr2.
3. Display the present value of ptr1 and ptr2.
4. Return to called place.

Program:

```
#include <stdio.h>
#include <conio.h>
void swap(int *, int *);

void main()
{
  int a = 10;
  int b = 20;
  clrscr();

  printf("From Main \n Before Swapping the Values \n a = %d, b = %d\n",a,b);
  swap(&a,&b);

  printf("\n From Main \n After Swapping Values at Function Swap a = %d, b = %d\n", a, b);
  getch();
}

void swap (int *ptr1, int *ptr2)
{
  int temp;
  printf("\nInside Function Swap");

  temp = *ptr1;
  *ptr1=*ptr2;
  *ptr2=temp;

   printf("\nAfter swapping values inside Function Swap");
   printf("\n *ptr1 = %d, *ptr2 = %d\n",*ptr1,*ptr2);
   printf("Exiting from Function Swap\n");

}
```

Sample Input / Output:

```
From Main
Before Swapping the Values
a = 10, b = 20

Inside Function Swap
After Swapping Values inside Function Swap
 *ptr1 = 20, *ptr2 = 10
Exiting from Function Swap

From Main
After Swapping Values at Function Swap
a= 20, b = 10
```

Result:

Thus, the C program to swap the contents available in two variables using function call by reference is executed successfully.

Ex. No. : 08 **Factorial of a Number using Recursive Functions**
Date :

Aim:

To find the factorial of a given number using recursive function in C language

Algorithm:

```
1. Start
2. Get the integer value for the variable n.
3. Call the function multiply(n) and assign the return value to the integer variable f .
4. Display the factorial of the entered value from the variable f.
5. Stop
```

Function factorial(n)

```
1. If (n>=1) then return (n *factorial(n-1))
          else return the value 1to called place
```

Program:

```
#include<stdio.h>
#include<conio.h>
long int factorial(int n);
void main()
{
 int n;
 long int f;
 clrscr();
 printf("Enter a positive integer: ");
 scanf("%d",&n);
 f=factorial(n);
 printf("Factorial of %d = %ld", n, f);
 getch();
}

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

Sample Input / Output:

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

Enter a positive integer: 6
Factorial of 5 is 720
```

Result:

Thus, the C program to find the factorial of a given number using recursive function is executed successfully.

Ex. No. : 09 **Problems involving Pointers and Structures**
Date :

Ex. No. 9(a) Pointers

Aim:

To understand the concept of pointers and its usage in C language

Algorithm:

1. Start.
2. Create a integer variable c and a integer pointer ptrc.
3. Assign 22 to the variable c.
4. Print the address and content of the variable c
5. Assign address of c to pointer ptrc.
6. Print the address and content of address available at ptrc.
7. Change the value c to 11.
8. Print the address and content of address available at ptrc.
9. Change the value c to 2 through the pointer ptrc
10. Print the address and content of the variable c
11. Stop.

Program:

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

void main()
{
  int* ptrc, c;
  clrscr();

  c = 22;
  printf("Address of c: %p\n", &c);
  printf("Value of c: %d\n\n", c);

  ptrc = &c;
  printf("Address available at pointer ptrc: %p\n", ptrc);
  printf("Content of Address available at pointer ptrc: %d\n\n", *ptrc);

  c = 11;
  printf("Address available at pointer ptrc: %p\n", ptrc);
  printf("Content of Address available at pointer ptrc: %d\n\n", *ptrc);

  *ptrc = 2;
  printf("Address of c: %p\n", &c);
  printf("Value of c: %d\n\n", c);
  getch();
}
```

Sample Input / Output:

```
Address of c: 26784
Value of c: 22

Address available at pointer ptrc: 26784
Content of Address available at pointer ptrc: 22

Address available at pointer ptrc: 26784
Content of Address available at pointer ptrc: 11

Address of c: 26784
Value of c: 2
```

Result:

Thus, the C program to understand the concept of pointers is executed successfully.

Ex. No. 9(b) **Structures**

Aim:

To calculate the Body Mass Index (BMI) of a person from the details stored in structure data type in C language

Algorithm:

```
1. Start
2. Create a structure person with the following members
        name[30] of character data type
        height  and weight of float data type
        bmi of double data type
3. Create array of structure p[10] of person data type.
4. Get the number of persons in the variable n.
5. Get the details of n persons in the structure array p[i] for all i such that 0 < = i < n.
6. Calculate the bmi of each person by using the formula bmi = weight/height$^2$ and store
   it in the corresponding bmi
7. Display the all the details of each person.
8. Stop
```

Program:

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

Extern _floatconvert;
#pragma extref _floatconvert

struct person
{
 char name[30];
 float height;
 float weight;
 double bmi;
};

void main()
{
 int i,n;
 struct person p[5];
 clrscr();
 printf("Enter the number of persons detail to be entered : ");
 scanf("%d", &n);
 for(i=0;i<n;i++)
 {
  printf("Enter the name of the Person : ");
  scanf("%s", p[i].name);

  printf("Enter the height of the Person(in meter) : ");
  scanf("%f", &p[i].height);

  printf("Enter the weight of the Person(in Kgs) : ");
  scanf("%f", &p[i].weight);
  p[i].bmi=p[i].weight/(p[i].height*p[i].height);
 }
 printf("\nPerson Details\n");
 printf("Name \t\t\t\t Height \t weight \t BMI\n");
 for(i=0;i<n;i++)
 {
  printf("%-30s \t %.2f \t\t %.2f \t\t%.2f \n", p[i].name, p[i].height, p[i].weight, p[i].bmi);
 }
 getch();
}
```

Sample Input / Output:

```
Enter the number of persons detail to be entered : 2

Enter the name of the Person : Kanthimathinathan
Enter the height of the Person(in meter) : 1.73
Enter the weight of the Person(in Kgs) : 66

Enter the name of the Person : Ramachandran
Enter the height of the Person(in meter) : 1.69
Enter the weight of the Person(in Kgs) : 67

Person Details

Name                        Height      Weight           BMI
Kanthimathinathan           1.73        66.00            22.05
Ramachandran                1.69        67.00            23.46
```

Result:

Thus, the C program to calculate the Body Mass Index (BMI) of a person from the details stored in structure data type is executed successfully.

Ex. No. : 10 **Problems involving File Operations**

Date :

Ex. No. 10(a) **Creation of a Simple Text File**

Aim:

To create a simple text file with contents in C Language.

Algorithm:

```
1. Start
2. Create a file and open it in writing mode with the file pointer fp.
3. If the file pointer fp is equal to NULL then display "Permission Denied" and go to
   Step 6.
4. Copy the content to the file through pointer fp.
5. Close the file.
6. Stop
```

Program:

```
#include <stdio.h>
#include <conio.h>
void main()
{
        FILE *fp = fopen("new1.txt", "w");
        clrscr();
        if(fp==NULL)
        {
          printf("\n Permission Denied to Create File \n");
          getch();
          exit(0);
        }
        else
        {
          fprintf(fp, "Hello Students.\nHave a NICE DAY ");
          printf("\n File Created Successfully");
          getch();
          fclose(fp);
        }
        getch();
 }
```

Sample Input / Output:

```
File Created Successfully
```

*[**Note:** After successful execution of this program, go to the dos prompt and execute the below command to display the contents of the file created]*

```
D:\>type new1.txt
Hello Students.
Have a NICE DAY
```

Result:

Thus, the C program to create a simple text file is executed successfully.

Ex. No. 10(b) Copying the Contents of one Text File into another Text File

Aim:

To copy the contents of one text file into another text file using C language

Algorithm:

1. Start
2. Create two file pointers fp1 and fp2.
3. Open the existing file using file pointer fp1 in reading mode.
4. Create and open a new file using the file pointer fp2 in writing mode.
5. If the file pointer fp1 is equal to NULL display “File Not Found” and go to Step 9.
6. If the file pointer fp2 is equal to NULL display “Permission Denied” and go to Step 9.
7. Read the content from the source file character by character pointed by fp1 and write it to the new file pointed by fp2 until end of source file.
8. Close both the files.
9. Stop

Program:

```
#include <stdio.h>
#include <conio.h>
void main()
{
   FILE *fp1, *fp2;
   char ch;

   fp1 = fopen("new1.txt", "r");
   fp2 = fopen("new2.txt", "w");
   clrscr();

    if(fp1==NULL)
    {
     printf(“\n File is not Present in that Location \n”);
     exit(0);
     }

    if(fp2==NULL)
    {
     printf(“\n Permission Denied to create File \n”);
     exit(0);
     }
     else
    {
     while(  (ch = getc(fp1) ) != EOF)
     {
       putc(ch, fp2);
      }
     }
     fclose(fp1);
     fclose(fp2);
     printf(“\nFile Copied Successfully”);
     getch();
}
```

Sample Input / Output:

```
File Copied Successfully
```

***[Note:** After successful execution of this program, go to the dos prompt and execute the below command to display the contents of the source and destination files.]*

```
D:\>type new1.txt

Hello Students
Have a NICE DAY

D:\>type new2.txt

Hello Students
Have a NICE DAY
```

Result:

Thus, the C program to copy the contents of one text file into another text file is executed successfully.

Ex. No. : 11 **Programming for Solving Problems on Numerical Methods**
Date :

Root Finding by Bisection Method

Aim:

To find the root of the given equation $f(x)=x^3+x^2-2$ using bisection method in C Language

Algorithm:

1. Start.
2. Get the range in which the root may be present i.e. assumed upper and lower boundary values [a, b] for the given equation $f(x) = x^3+x^2-2$
3. If (func(a)*func(b)>=0) then Display “You have not entered correct boundary values”
4. Get the precision value E
5. While the difference between b and a is greater than the given precision value E repeat the following steps
    a. Find middle point c= (a + b)/2.
    b. If f(c) == 0, then display c is the root of the solution and go to step 6.
    c. If f(a)*f(c) < 0 then change upper boundary as c.
    d. If f(b)*f(c) < 0 then change lower boundary as b.
6. Stop

Program:

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

double E;
double func(double x)
{
   return x*x*x - x*x + 2;
}

void bisection(double a, double b)
{
   double c = a;
   if (func(a) * func(b) >= 0)
   {
         printf("You have not entered correct boundary values \n");
         return;
   }
   while ((b-a) >= E)
   {
         // Find middle point
         c = (a+b)/2;
         // Check if middle point is root
         if (func(c) == 0.0)
            break;

         // Decide the side to repeat the steps
         else if (func(c)*func(a) < 0)
            b = c;
            else
                a = c;
    }
    printf("The value of root is : %f", c);
}

void main()
{
    double a, b;
    clrscr();
    printf("Enter the Range in which the Root may present \n");
    printf("\nEnter the Lower bound :");
    scanf("%lf",&a);
    printf("Enter the Upper bound :");
    scanf("%lf",&b);
    printf("Enter the precision value :");
    scanf("%lf",&E);
    bisection(a, b);
    getch();
}
```

Sample Input / Output:

```
Enter the Range in which the Root may present

Enter the Lower bound : -200
Enter the Upper bound : 300
Enter the precision value : 0.000001

The value of root is  : -1.000000

Enter the Range in which the Root may present

Enter the Lower bound : -200
Enter the Upper bound : 300
Enter the precision value : 0.0001

The value of root is  : -1.000059

Enter the Range in which the Root may present

Enter the Lower bound : 1000
Enter the Upper bound : 2000
Enter the precision value : 0.0001

You have not entered correct boundary values
```

Result:

Thus, the C program to find the root of the given equation $f(x)=x^3+x^2-2$ using bisection method is executed successfully.