VIJAYAM INSTITUTE OF TECHNOLOGY

Sundararajapuram (V), G.D.Nellore, Opp. Heritage Dairy, Puttur Road,
Chittoor Dist-517125

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Anantapuramu)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Introduction to Programming Lab Manual (R23)

VIJAYAM INSTITUTE OF TECHNOLOGY

Sundararajapuram (V), G.D.Nellore, Opp. Heritage Dairy, Puttur Road,
Chittoor Dist-517125

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Anantapuramu)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Introduction to Programming Lab Manual (R23)

Name:____

Reg. No:____

Year/Semester:_______________________________________

Academic Year:_______________________________________

INDEX

Sl.No	NAME OF THE EXPERIMENT	Date	Page No	Marks obtained	Signature of the faculty

INTRODUCTION TO PROGRAMMING (RECORD)

Developing the algorithms forSum and average of 3 numbers

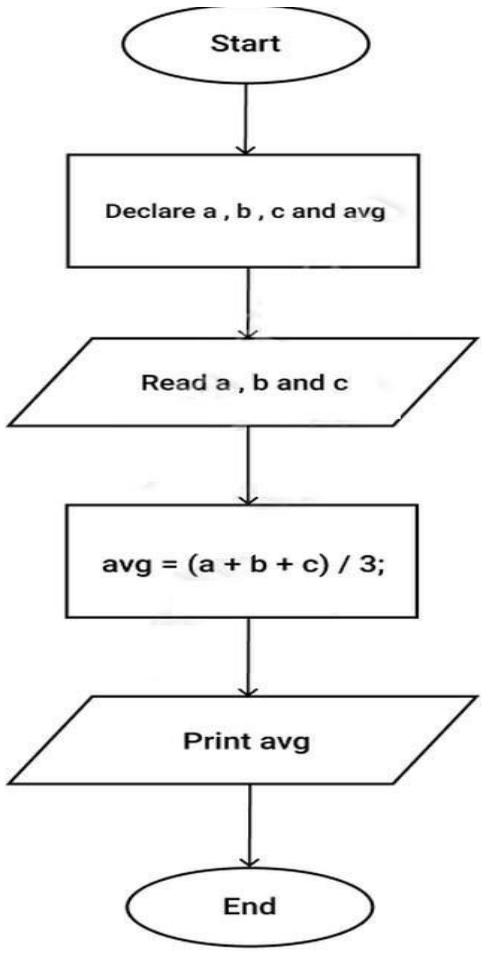
AIM:

To developing the algorithms/flowcharts for Sum and average of 3 numbers.

ALGORITHM:

1. Start the program.
2. Declare the variables in all data types and initialize it.
3. Calculating the average. Say the numbers are a,b,c..
4. Add the all values of a, b and c. Then average like **avg = sum / 3.**
5. Print the result.
6. Stop the program

FLOW CHART:



PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
 {
        int a, b, c ,sum;
        float avg;     //Declaring the variables
        clrscr();
        printf("Enter a three values: \n");
        scanf("%d%d%d", &a,&b,&c);    //Reading Variables a,b,c
        sum=a+b+c;
        printf(" sum of three numbers :%d \n”,sum);
        avg =  sum / 3;
        printf(" Average of three number: %f", avg);
        getch();
}
```

OUTPUT:

```
Enter three values: 10 20 30
sum of three numbers : 60
Average of three number: 20.0
```

Result:

Thus the program was compiled and executed successfully

Developing the algorithms for Conversion of Fahrenheit to Celsius and vice versa

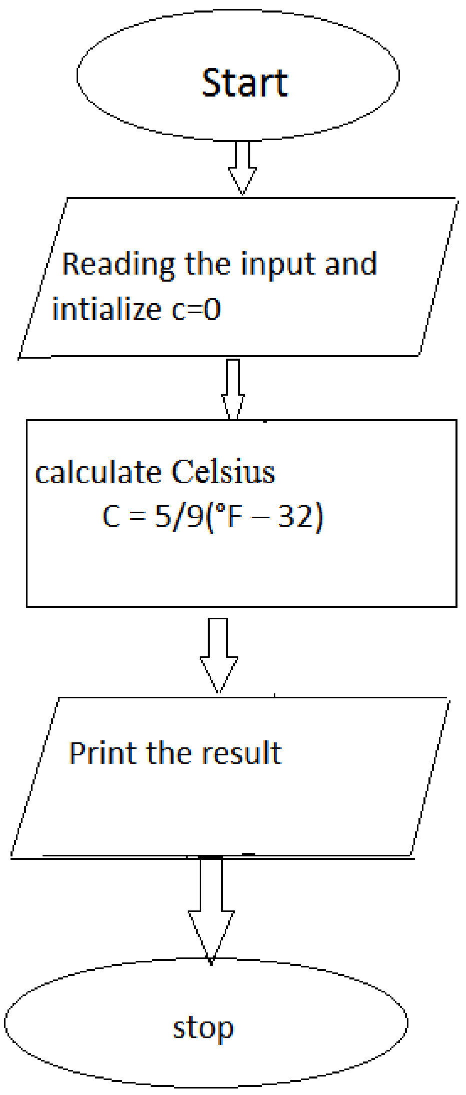
AIM:

To developing the algorithms/flowcharts for Conversion of Fahrenheit to Celsius.

ALGORITHM:

1. Start the program.
2. Declare the variables in all data types and initialize it.
3. Converting the temperature Fahrenheit to Celsius
4. Get the input as Fahrenheit f and calculate Celsius °C = 5/9(°F – 32)
5. Print the result.
6. Stop the program.

FLOW CHART:


Start
Reading the input and intialize c=0
calculate Celsius C = 5/9(°F – 32)
Print the result
stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int  fahremheit, celcius;
        clrscr();
        printf("Enter the value of Fahrenheit:\n");
        scanf("%d", &fahrenheit);
        celcius=(fahrenheit-32)*5 /9;
        printf("Celcius is :%d" ,c);
         getch();

}
```

OUTPUT:

```
Enter the value of Fahrenheit: 75
Celcius is : 23
```

Result:

Thus the program was compiled and executed successfully.

Developing the algorithms forSimple interest Calculation

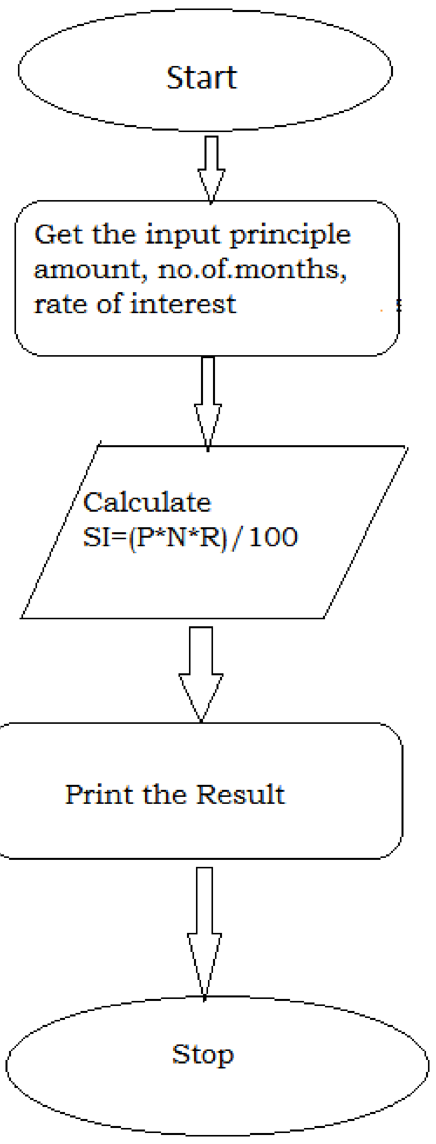
AIM:

To developing the algorithms/flowcharts for Simple interest Calculation.

Algorithm:

1. Start the program.
2. Declare the variables in all data types and initialize it.
3. Get the input principal amount, TIME IN YEAR, rate of interest.
4. Calculate Simple Interest=(p*t*r)/100.
5. Print the result.
6. Stop the program

FLOW CHART:


Start
Get the input principle amount, no.of.months, rate of interest
Calculate SI=(P*N*R)/100
Print the Result
Stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
      float principal, time, rate, si;
      printf(“enter the principal, time, and rate:”);
      scanf(“%d%d%d”, &principal, &time, &rate);
      si= (principal*time*rate)/100;
      printf(“Simple interest calculation is: %f”,si);
      getch();
 }
```

OUTPUT:

```
enter the principal, time, and rate:1200
2
54
Simple interest calculation is :1296.000
```

Result:

Thus the program was compiled and executed successfully

Finding the Square Root of a Given Number

AIM:

To Write a C Program finding the square root of a given number

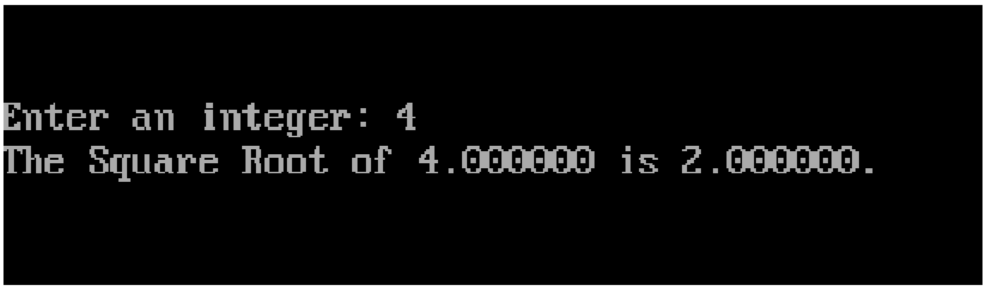
Algorithm:

1. Start the program.
2. Declare the variables and initialize it.
3. Calculating the square root value.
4. Print the result.
5. Stop the program.

Program:

```
#include
<stdio.h>
#include
<math.h>      void
main()
{
        float num, root;
        printf("Enter an integer:
        "); scanf("%f", &num);
        root = sqrt(num);
        printf("The Square Root of  %f is %f:", num, root);
        getch();
}
```

OUTPUT:



```
Enter an integer: 4
The Square Root of 4.000000 is 2.000000.
```

Result:

Thus the
program
was
compiled
and
executed
successfully

Finding compound interest

AIM:

To find the compound interest of the given parameters.

ALGORITHM:

1. Start the program
2. Define variable p(principal),t(time),r(rate) and compound interest.
3. Get the input value from the user
4. A=p(1+r/100)*t
5. Calculate the compound interest using the result=amount-p formula
6. stop the program

PROGRAM:

```
#include<stdio.h
>
#include<conio.h
> void main()
{
    float p,t,r,ci;
    printf("Enter
    principal:");
    scanf("%f",&p);
    printf("Enter rate value:");
    scanf("%f",&r);
    printf("Enter time:");
    scanf("%f ".&t);
    ci=p*pow((1+r/100)*t);
    printf("compound interest is %f", ci);
    getch();
}
```

OUTPUT:

```
Enter principal: 1200
Enter  rate:2
Enter time:5
Compound interest is 1324.896973
```

Result:

Thus the program was compiled and executed successfully

Area of a triangle using heron’s formulae

AIM: To Write a C Program finding Area of a triangle using heron’s formulae.

Algorithm:

1. Start the program.
2. Declare the variables in all data types and initialize it.
3. Get the input from user.
4. Steps to find the area of a triangle using Heron's formulaLet A, B and C be the length of three sides of a triangle.
5. Calculate the semi perimeter of the triangle.

   Semi-Perimeter of triangle(S) = (A + B + C)/2
6. Now, we can calculate the area of triangle using below mentioned formula.

   Area of Triangle = √ S(S-A)(S-B)(S-C))

   Where, S is the semi-perimeter that we calculated in first step
7. Print the result.
8. Stop the program.

ALGORITHM:

```
#include <stdio.h>
#include <math.h>
#include<conio.h>
void main()
{
        float a, b, c, s, area;
        printf("Enter the length of three sides of triangle:\n");
        scanf("%f %f %f", &a,&b, &c);
        s = (a+b+c)/2;
        area = sqrt(s*(s-a)*(s-b)*(s-c));
        printf("Area of triangle : %0.4f\n", area);
        getch();
}
```

OUTPUT:

```
Enter the length of three sides of triangle:
12  7  9
Area of triangle :  31.3050
```

RESULT:

Thus the program was executed successfully.

Distance travelled by an object

AIM:

To Write a C Program finding Distance travelled by an object.

Algorithm:

1. Start the program.
2. Declare the variables in all data types and initialize it.
3. Get the input value velocity, acceleration and time from user.
4. Calculate the Distance travelled by an object by using this formule
5. $D=ut+at^2$ where u- velocity, a-acceleration and t-time
6. Print the result.
7. Stop the program.

ALGORITHM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
         float u, a, d;
        int t;
        clrscr();
        printf("Enter the value of acceleration : ");
        scanf("%f \n", & a);
        printf("Enter the value of Velocity: ");
        scanf("%f \n", & u);
        printf("Enter the value of time : ");
        scanf("%d \n", & t);
        d = (u * t) + (a * t * t) / 2;
        printf("The Distance travelled by the object : %.2f", d);
        getch();
}
```

OUTPUT:

```
Enter the value of acceleration :  6
Enter the value of Velocity:  7
Enter the value of time :  8
The Distance travelled by the object :  248.00
```

Result:

Thus the program was compiled and executed successfully

OPERATOR PRECEDENCE AND ASSOCIATIVITY

AIM:

To write c program Evaluate the following expresions.

a. A+B*C+(D*E)+F*G
b. A/B*C-B+A*D/3
c. A++ +B---A
d. J=(i++)+(++i)

ALGORITHM:

1. Start the program
2. Declare the variable
3. Get the value of the variable from the user
4. Evaluate the following expressions
    a. A+B*C+(D*E)+F*G
    b. A/B*C-B+A*D/3
    c. A++ +B---A
    d. J=(i++)+(++i)
5. Print the result
6. Stop the program.

Program:

```
#include<stdio.h>
#include<conio.h>
Void main()
{
int A,B,C,D,E,F,G,i;
int res,res1,res2,res3;
clrscr();
printf("Enter the value of A:");
Scanf("%d \n" ,&A);
printf("Enter the value of B:");
Scanf("%d \n" ,&B);
printf("Enter the value of C:");
Scanf("%d \n" ,&C);
printf("Enter the value of D:");
```

```
        Scanf("%d \n" ,&D);
        printf("Enter the value of E:");
        Scanf("%d \n" ,&E);
        printf("Enter the value of F:");
        Scanf("%d \n" ,&F);
        printf("Enter the value of G:");
        Scanf("%d \n" ,&G);
        printf("Enter the value of  i:");
        Scanf("%d \n" ,&i);
        res=A+B*C+(D*E)+F*G;
        Printf(" The result of the expression  one  : %d \n",res);
        res1=A/B*C-B+A*D/3;
        Printf(" The result of the expression  two: %d \n",res1);
        res2=A++ +B---A;
        Printf(" The result of the expression  three :%d \n" ,res2);
        res3=(i++)+(++i);
        Printf(" The result of the expression four : %d \n",res3);
        getch();
    }
```

OUTPUT:

```
Enter the value of A:  4
Enter the value of B:  5
Enter the value of C:  6
Enter the value of D:  2
Enter the value of E:  5
Enter the value of F:  7
Enter the value of G:  2
Enter the value of  i:  2
The result of the expression  one  :  58
The result of the expression  two  :  -3
The result of the expression  three  :  5
The result of the expression four :   6
```

Result:

Thus the program was compiled and executed successfully.

Find the maximum of three numbers using conditional operators

AIM:

To write c program find the maximum of three numbers using conditional operators.

ALGORITHM:

1. Start the program.
2. Declare the variables.
3. Get the input value from the user.
4. Find the maximum of three numbers using conditional operator.
5. Check the condition if(a>b && a>c) then print a
6. else if(b>c && b>a) then print b
7. else print c
8. Stop the program.

Program:

```
# include<stdio.h>
#include<conio.h>
void main()
{
         int a,b,c;
        clrscr();
        printf("enter a,b,c values:");
        scanf("%d%d%d",&a,&b,&c);
        if(a>b && a>c)
        {
                printf("  a is biggest number : %d",a);
        }
        else if(b>c && b>a)
        {
                Printf(" b is  biggest  number : %d",b);
        }
        else
        {
                Printf(“ c is biggest number :%d”,c);
        }

        getch();
}
```

OUTPUT:

```
enter a,b,c values: 10 20 30
c is biggest number : 30
```

RESULT:

Thus the program was compiled and executed successfully.

Type Conversion

AIM:

To Write C Program take marks of 5 subjects in integers, and find the total, average in float.

ALGORITHM:

1. Start the program.
2. Declare the variables.
3. Get the input from user.
4. Calculate the total, average of five subject marks.
5. Print the result.
6. Stop the program

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int sub1,sub2,sub3,sub4,sub5;
float avg, total ;
clrscr();
printf("Enter the subject one marks : ");
scanf("%d\n",&sub1);
printf("Enter the subject two marks : ");
 scanf("%d\n",&sub2);
printf("Enter the subject three marks : ");
scanf("%d\n",&sub3);
printf("Enter the subject four marks : ");
scanf("%d\n",&sub4);
printf("Enter the subject five marks : ");
scanf("%d\n",&sub5);
tot=(sub1+sub2+sub3+sub4+sub
5); avg=tot/5;
printf("The student Total is : %f \n",tot);
printf("The student Average: %f \n",avg);
getch();
}
```

OUTPUT:

```
Enter the subject one marks :  55
Enter the subject two marks :
65 Enter the subject three marks
: 75 Enter the subject four marks
:  85 Enter the subject five marks
: 95 The student Total is :
The student Average :
```

RESULT:

Thus the program was compiled and executed successfully.

EX.NO:5(i)

Find the Max and Min of Numbers Using If-Else

Date:

Aim: To Write a C program to find the max and min of four numbers using if-else.

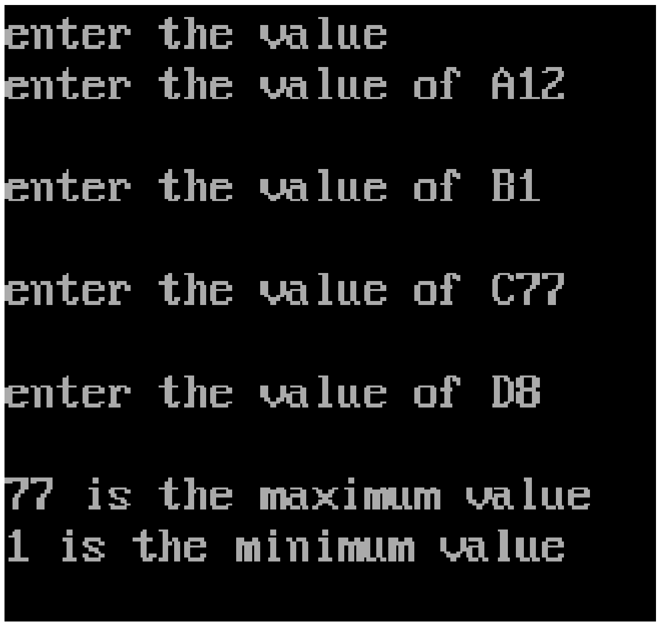
Algorithm:

1. Start the program.
2. Declare the variables.
3. Get the value of the variables from the user.
4. Find the Max and Min value of the getting Numbers Using If-Else statements.
5. Print the result.
6. Stop the program.

Program:

```
#include<stdio.h
>
#include<conio.h
>int a,b,c,d;
void main()
{
int
max,min;
clrscr();
printf("enter the value");
printf("\nenter the value of
A");scanf("%d",&a);
printf("\nenter the value of
B");scanf("%d",&b);
printf("\nenter the value of
C");scanf("%d",&c);
printf("\nenter the value of
D");scanf("%d",&d);
max=min=
a;
if(max<b)
{ max=b;}
else
if(b<min)
{min=b;
}
if(max<c
)
{ max=c;}
else
if(c<min)
{min=c;
}
if(max<d
)
{ max=d;}
else
if(d<min)
{min=d;}
printf("\n%d is the maximum
value",max);printf("\n%d is the
minimum value",min);

getch();
}
```

Output:



```
enter the value
enter the value of A12

enter the value of B1

enter the value of C77

enter the value of D8

77 is the maximum value
1 is the minimum value
```

Result:

Thus the program was compiled and executed successfully

EX.NO:5(ii)	**Generate Electricity Bill**

Date:

Aim:

To Write a C program to generate electricity bill

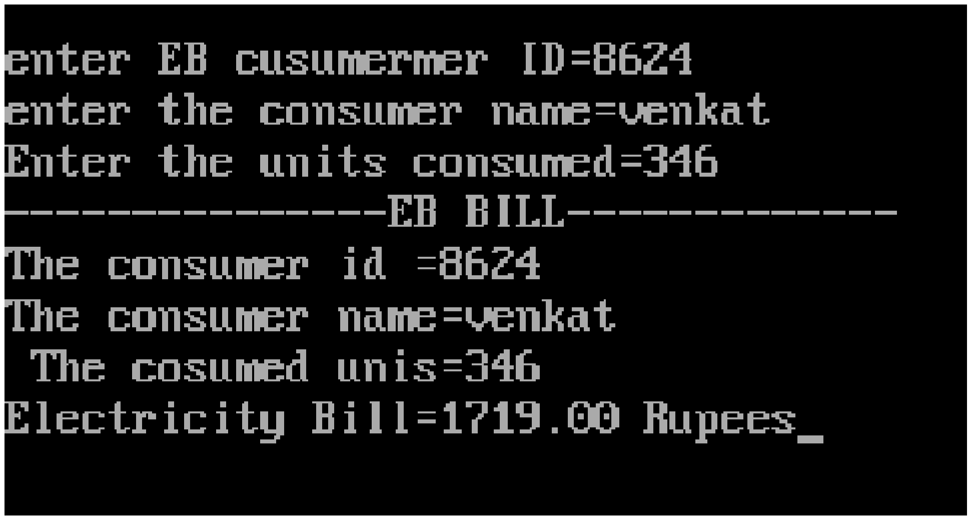
Algorithm:

1. Start the program.
2. For example the conditions are
   For first 50 units – Rs. 3.50/unit
   For next 100 units – Rs. 4.00/unit
   For next 100 units – Rs. 5.20/unit
   For units above 250 – Rs. 6.50/unit You can use conditional statements.
3. Declare the variables.
4. Get the input unit consumed by customer in some variable say unit value from the user.
5. Calculate EB Bill using the above conditions.
6. Print the result.
7. Stop the program

Program:

```
#include<stdio.h
>
#include<conio.h
>int main()
{
	float bill;
	int
	id,units;
	char
	name[20];
	clrscr();
	printf("enter EB cusumermer
	ID=");scanf("%d",&id);
	printf("enter the consumer
	name=");scanf("%s",name);
	printf("Enter the units
	consumed=");
	scanf("%d",&units);
	printf("....................EB BILL
	.................................");printf("\nThe
	consumer id =%d",id); printf("\nThe
	consumer name=%s",name);printf("\n
	The cosumed unis=%d",units);
	if(units<=50 && units>=0)
	{
		bill=units*3.50;
		printf("\nElectricity Bill=%.2f Rupees",bill);
	}
	else if(units<=100 && units>50)
	{
		bill=50*3.50+(units-50)*4;
		printf("\nElectricity Bill=%.2f
		Rupees",bill);
	}
	else if(units<=250 && units>150)
	{
		bill=50*3.50+100*4+(units-150)*5.20;
		printf("\nElectricity Bill=%.2f
		Rupees",bill);
	}
	else if(units>250)
	{


			}
	}
	els
	e
	{
  bill=50*      printf("\nElectricity Bill=%.2f Rupees",bill);
  3.50+10
  0*4+10
  0*5.20+
  (units-       printf("\nPlease enter valid consumed units...");
  250)*6.
  50;
        getch();
        return
        0;
}
```

Output:



```
enter EB cusumermer ID=8624
enter the consumer name=venkat
Enter the units consumed=346
---------------EB BILL-------------
The consumer id =8624
The consumer name=venkat
 The cosumed unis=346
Electricity Bill=1719.00 Rupees_
```

Result:

Thus the program was compiled and executed successfully

EX.NO:4(iii)	Roots of the Quadratic Equation
Date:	

Aim:

To write C Program find the roots of the quadratic equation.

Algorithm:

1. Start the program.
2. Initialize all the variables used in the quadratic equation.
3. Get the input of all coefficient variables x, y and z from the user.
4. And then, find the discriminant of the quadratic equation using the formula:

   Discriminant = (y * y) - (4 * x *z).
5. Calculate the roots based on the nature of the discriminant of the quadratic equation.
6. If discriminant > 0, then

   Root1 = (-y + sqrt(det)) / (2 * x)

   Root2 = (-y + sqrt(det)) / (2 * x)

   Print the roots are real and distinct.
7. Else if (discriminant = 0) then,

   Root1 = Root2 = -y / (2 * x).

   Print both roots are real and equal.
8. Else (discriminant < 0), the roots are distinct complex where,

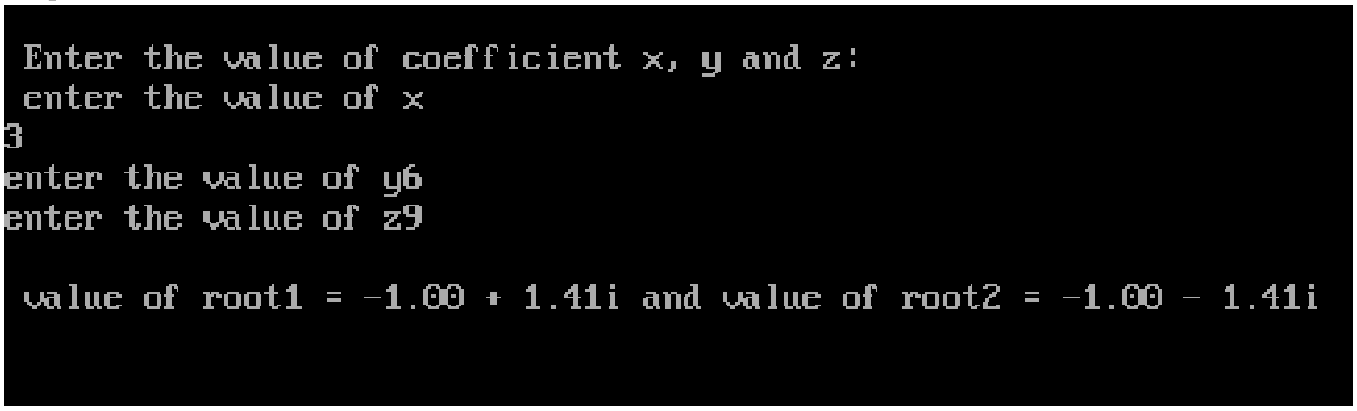
   Real part of the root is: Root1 = Root2 = -y / (2 * x) or real = -y / (2 * x).Imaginary part of the root is: sqrt(-discriminant) / (2 * x).

   Print both roots are imaginary, where first root is (r + i) img and second root is (r - i)img.
9. Stop the program.

Program:

```
#include<stdio.h>
#include<math.h>
#include<conio.h
>void main()
{
   float x, y, z, det, root1, root2, real,
   img;clrscr();
   printf("\n Enter the value of coefficient x, y and z: \n
   ");printf("enter the value of x");
   scanf("%f",&x);
   printf("enter the value of
   y");scanf("%f",&y);
   printf("enter the value of
   z");scanf("%f",&z);
   det = y * y - 4 * x *
   z;if (det > 0)
   {
   root1 = (-y + sqrt(det)) / (2 *
   x);root2 = (-y + sqrt(det)) / (2
   * x);
   printf("\n Value of root1 = %.2f and value of root2 = %.2f", root1, root2);
   }
   else if (det == 0)
   {
      root1 = root2 = -y / (2 * x); // both roots are equal;
      printf("\n Value of root1 = %.2f and Value of root2 = %.2f", root1, root2);
   }
   else
   {
      real = -y / (2 * x);
      img = sqrt(-det) / (2 * x);
      printf("\n value of root1 = %.2f + %.2fi and value of root2 = %.2f - %.2fi ", real, img, real,
      img);
   }
   getch();
   }
```

Output:



```
 Enter the value of coefficient x, y and z:
 enter the value of x
3
enter the value of y6
enter the value of z9

 value of root1 = -1.00 + 1.41i and value of root2 = -1.00 - 1.41i
```

Result: Thus the program was compiled and executed successfully

EX.NO:4(iv)

Date:

Calculator Using Switch Case

Aim:

To Write a C program to simulate a calculator using switch case.

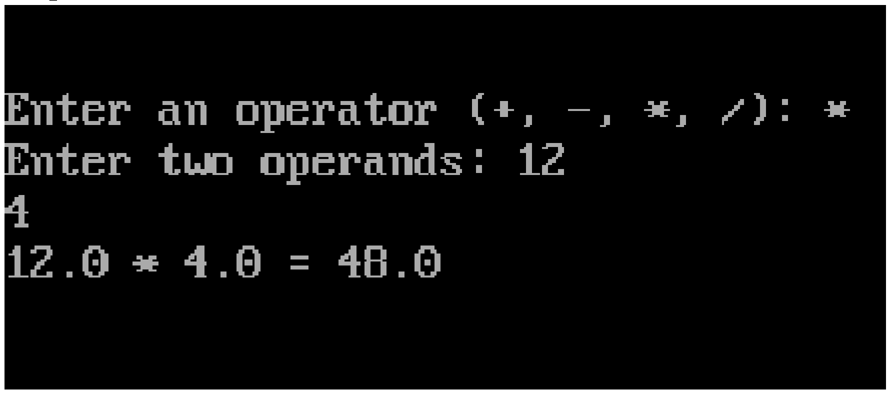
Algorithm:

1. Start the program.
2. Declare the variables.
3. Get the operator and values from the user.
4. Develop the calculator using switch case statements.
5. Generate the result based on user input.
6. Print the result.
7. Stop the program.

Program:

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

int main() {

  char op;
  double
  a,b;
  clrscr();
  printf("Enter an operator (+, -, *, /):
  ");scanf("%c", &op);
  printf("Enter two operands:
  ");scanf("%lf %lf", &a,
  &b);

  switch (op)
    {case '+':
      printf("%.1lf + %.1lf = %.1lf", a,b,
      a+b);break;
    case '-':
      printf("%.1lf - %.1lf = %.1lf", a,b,
      a-b);break;
    case '*':
      printf("%.1lf * %.1lf = %.1lf", a,b,
      a*b);break;
    case '/':
      printf("%.1lf / %.1lf = %.1lf", a,b,
      a/b);break;
    // operator doesn't match any case
    constantdefault:
      printf("Error! operator is not correct");
  }
  getch();
  return
  0;
}
```

Output:



```
Enter an operator (+, -, *, /): *
Enter two operands: 12
4
12.0 * 4.0 = 48.0
```

Result:

Thus the program was compiled and executed successfully

EX.NO:4(v)	**Find the Given Year is a Leap Year or Not.**
Date:	

Aim: To Write a C program to find the given year is a leap year or not.

Algorithm:

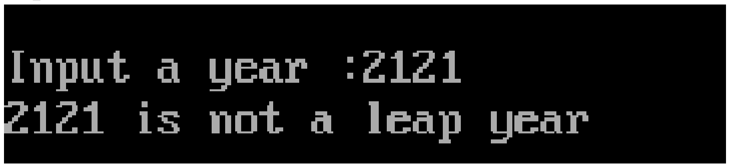
1. Start the program.
2. Declare the variables.
3. Get the input values from the user.
4. Find the given year is a leap year or not.
5. Print the result.
6. Stop the program.

Program:

```
#include
<stdio.h>
#include<conio.h
>void main()
{
   int
   year;
   clrscr()
   ;
   printf("Input a year
   :");scanf("%d",
   &year);
   if ((year % 400) == 0)
      printf("%d is a leap year.\n",
   year);else if ((year % 100) == 0)
      printf("%d is a not leap year.\n",
   year);else if ((year % 4) == 0)
      printf("%d is a leap year.\n",
   year);else
         printf("%d is not a leap year \n",
   year);getch();
}
```

d and executed successfully.

Output:



```
Input a year :2121
2121 is not a leap year
```

EX.NO:6(i)

Date:

Aim:

Result:

Thu
s
the
prog
ram
was
com
pile

Find the factorial of given number using any loop.

To Write a C program to find the factorial of given number using any loop.

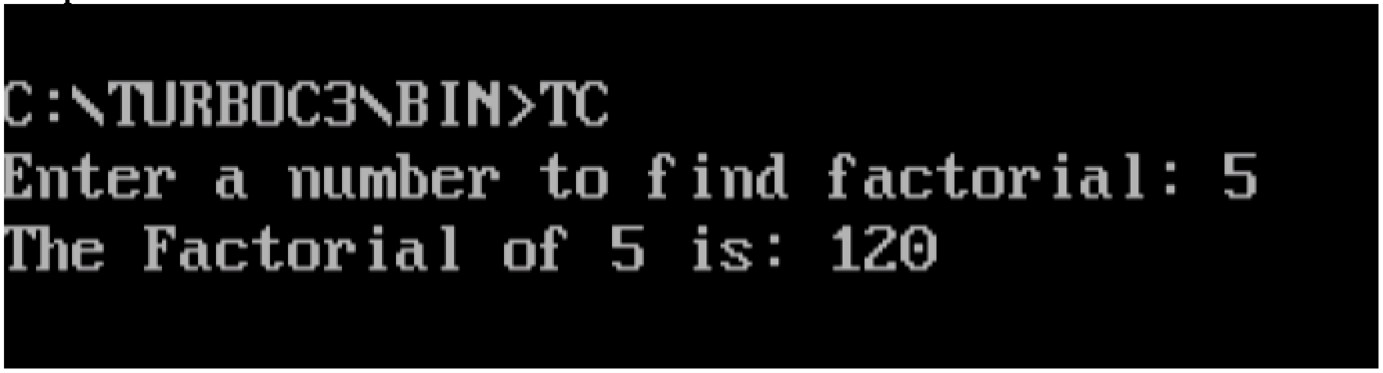
Algorithm:

1. Start the program.
2. Declare the variables.
3. Get the value of the variables from the user.
4. Find the factorial of given number using for loop.
5. Print the result.
6. Stop the program.

Program:

```
#include
<stdio.h>Void
main()
{
   int i,f=1,num;

  printf("Enter a number to find factorial: ");
  scanf("%d",&num);

  for(i=1;i<=num;i++)
 {
   f=f*i;
 }
printf("The Factorial of %d is: %d\n",num,f);
}
```

Output:



```
C:\TURBOC3\BIN>TC
Enter a number to find factorial: 5
The Factorial of 5 is: 120
```

Result:

Thus the program was compiled and executed successfully

EX.NO:6(ii)

Find the given number is a prime or not

Date:

Aim:

To Write a C program to find the given number is a prime or not

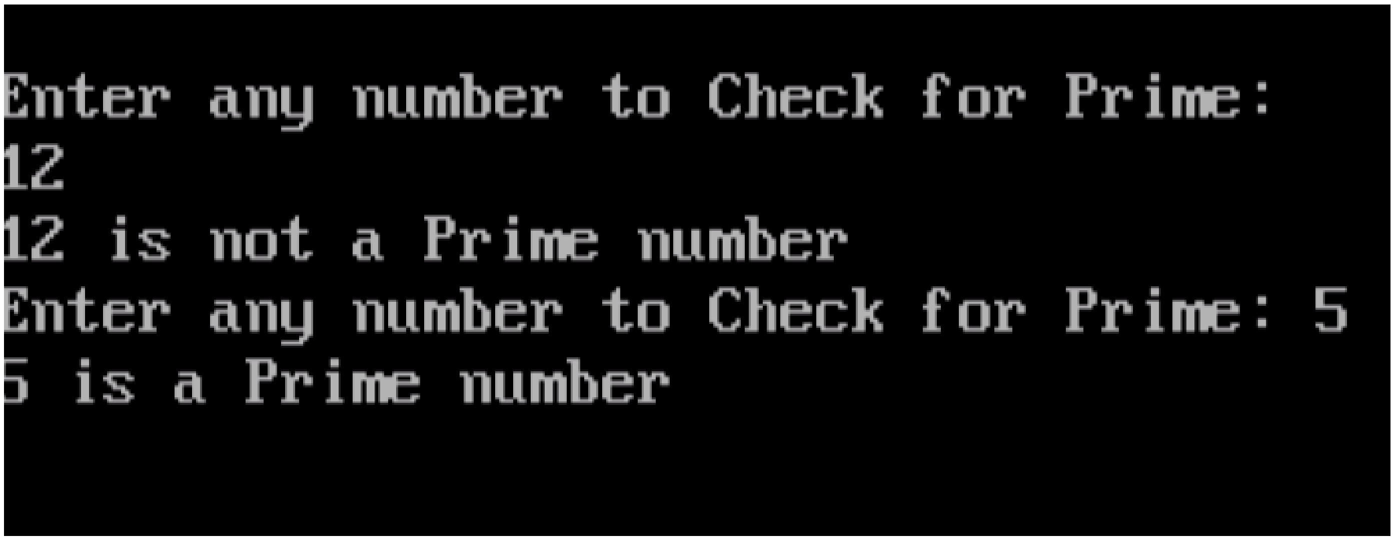
Algorithm:

```
1.  Start the program.
2.  Take num as input.
3.  Initialize a variable temp to 0.
4.  Iterate a "for" loop from 2 to num/2.
5.  If num is divisible by loop iterate, then increment temp.
6.  If the temp is equal to 0,
7.      Return "Num IS PRIME".
8.  Else,
9.      Return "Num IS NOT PRIME".
10. Print the result.
11. Stop the program
```

Program:

```
#include
<stdio.h>int
main()
{
   int i, num, temp = 0;
   printf("Enter any number to Check for
   Prime: ");scanf("%d", &num);
   for (i = 2; i <= num / 2; i++)
   {
      if (num % i == 0)
      {
         temp++
         ;break;
      }
   }
   if (temp == 0 && num != 1)
   {
     printf("%d is a Prime number", num);
   }
   else
   {
      printf("%d is not a Prime number", num);
   }
   return 0;
}
```

Output:



```
Enter any number to Check for Prime:
12
12 is not a Prime number
Enter any number to Check for Prime: 5
5 is a Prime number
```

Result:

Thus the program was compiled and executed successfully

EX.NO:6(iii)

Date:

Compute sine and cos series

Aim:

To write C Program to compute sine and cos series.

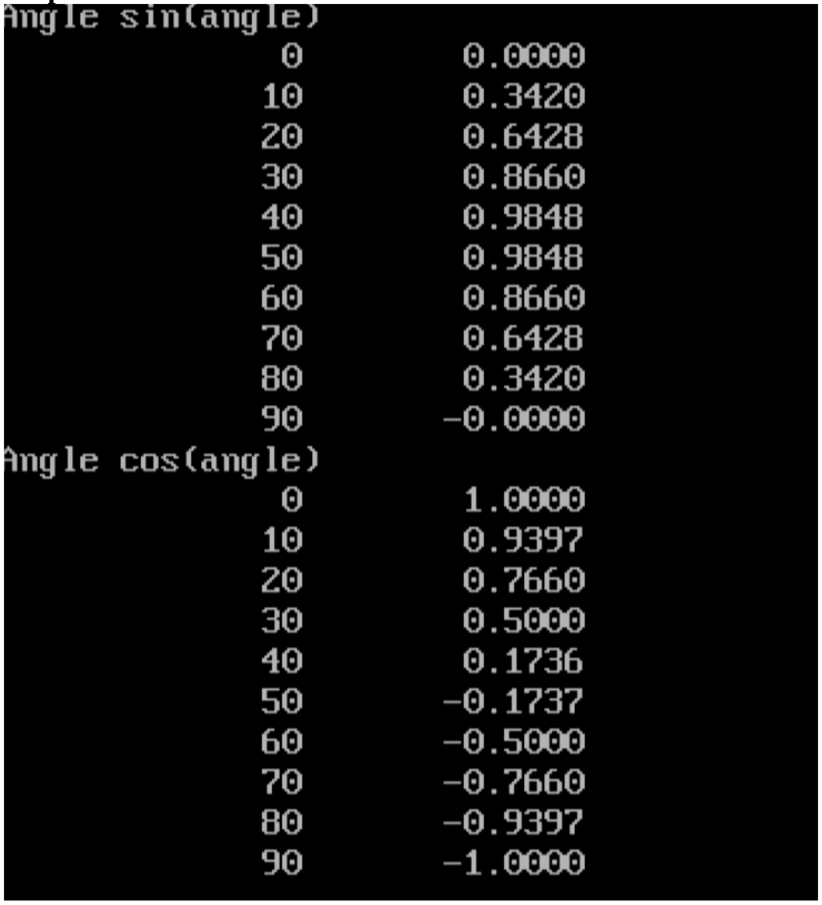
Algorithm:

1. Start the program.
2. Declare the variables.
3. Get the value of the variables from the user.
4. Compute sine and cos series.
5. Print the result.
6. Stop the program.

Program:

```
#include<stdio.h
>           #include
<math.h>#define
PI 3.1416
#define     MAX
150main ( ){
  int angle;
  float x,y;
  angle    =
  0;
  printf("Angle
  sin(angle)");while(angle
  <= MAX){
    x =
    (PI/MAX)*angle;y
    = sin(x);
    printf("%15d %13.4f", angle,
    y);angle = angle + 10;
  }
  printf("Angle
  cos(angle)");while(angle
  <= MAX)
  {
    x =
    (PI/MAX)*angle;y
    = cos(x);
    printf("%15d %13.4f", angle,
    y);angle = angle + 10;
  }
}
```

Output:



```
Angle sin(angle)
             0        0.0000
            10        0.3420
            20        0.6428
            30        0.8660
            40        0.9848
            50        0.9848
            60        0.8660
            70        0.6428
            80        0.3420
            90       -0.0000
Angle cos(angle)
             0        1.0000
            10        0.9397
            20        0.7660
            30        0.5000
            40        0.1736
            50       -0.1737
            60       -0.5000
            70       -0.7660
            80       -0.9397
            90       -1.0000
```

Result:

Thus the program was compiled and executed successfully

EX.NO:6(iv)

Date:

Checking a number palindrome

Aim:

To Write a C program to Checking a number palindrome.

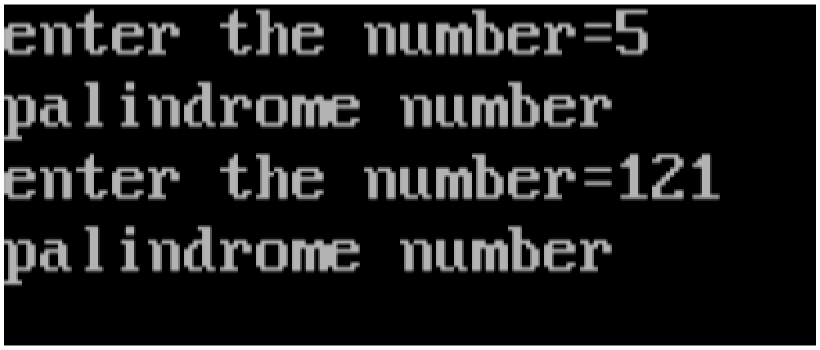
Algorithm:

1. Start the program.
2. Declare the variables.
3. Get the number from user
4. Hold the number in temporary variable
5. Reverse the number
6. Compare the temporary number with reversed number
7. If both numbers are same, print palindrome number
8. Else print not palindrome number
9. Print the result.
10. Stop the program.

Program:

```
#include<stdio.h
>int main()
{
int n,r,sum=0,temp;
printf("enter the
number=");
scanf("%d",&n);
temp=n;
while(n>0
)
{
r=n%10
;
sum=(sum*10)+
r;n=n/10;
}
if(temp==sum)
printf("palindrome number
");else
printf("not
palindrome");return 0;
}
```

Output:



```
enter the number=5
palindrome number
enter the number=121
palindrome number
```

Result:

Thus the program was compiled and executed successfully

EX.NO:6(v)	**Construct a pyramid of numbers.**
Date:	

Aim: To Write a C program to construct a pyramid of numbers.

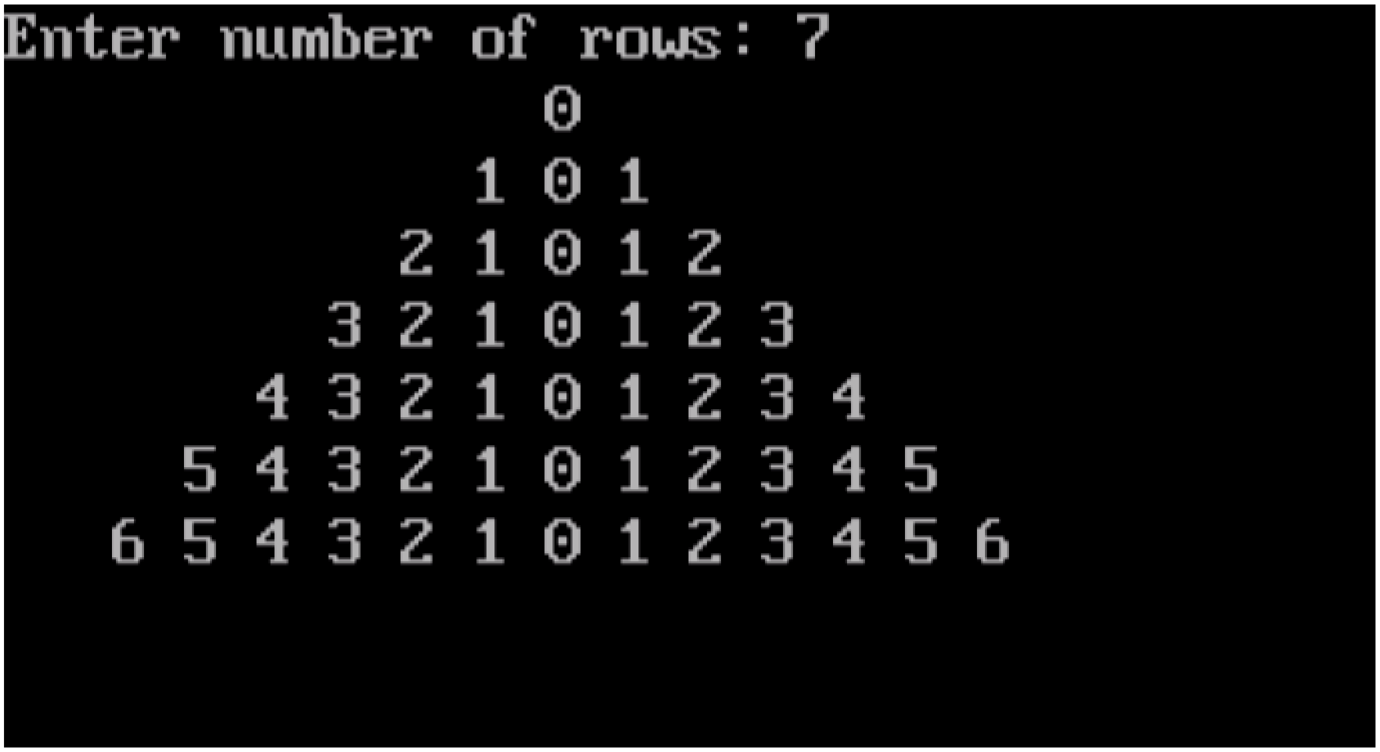
Algorithm:

1. Start the program.
2. Declare the variables.
3. Get the input values from the user.
4. To construct a pyramid of numbers.
5. Print the result.
6. Stop the program.

Program:

```
#include
<stdio.h>
#include<conio.h
>int main()
{
   int rows,  space, i,
   j;clrscr();
   printf("Enter number of rows:
   ");scanf("%d",&rows);

   for(i=0; i<rows; i++)
   {
      for(space=1; space <= rows-i;
         space++)printf(" ");

      for(j=0-i; j <= i; j++)
      {

         printf("%2d",abs(j));
      }
      printf("\n");
   }

}
```

Output: mpiled and executed successfully.



```
Enter number of rows: 7
              0
            1 0 1
          2 1 0 1 2
        3 2 1 0 1 2 3
      4 3 2 1 0 1 2 3 4
    5 4 3 2 1 0 1 2 3 4 5
  6 5 4 3 2 1 0 1 2 3 4 5 6
```

EX.NO:8(i)

Date:

Aim:

Result:

Th
us
th
e
pr
og
ra
m
wa
s
co

Addition of two matrices

To write C Program to perform Addition of two matrices.

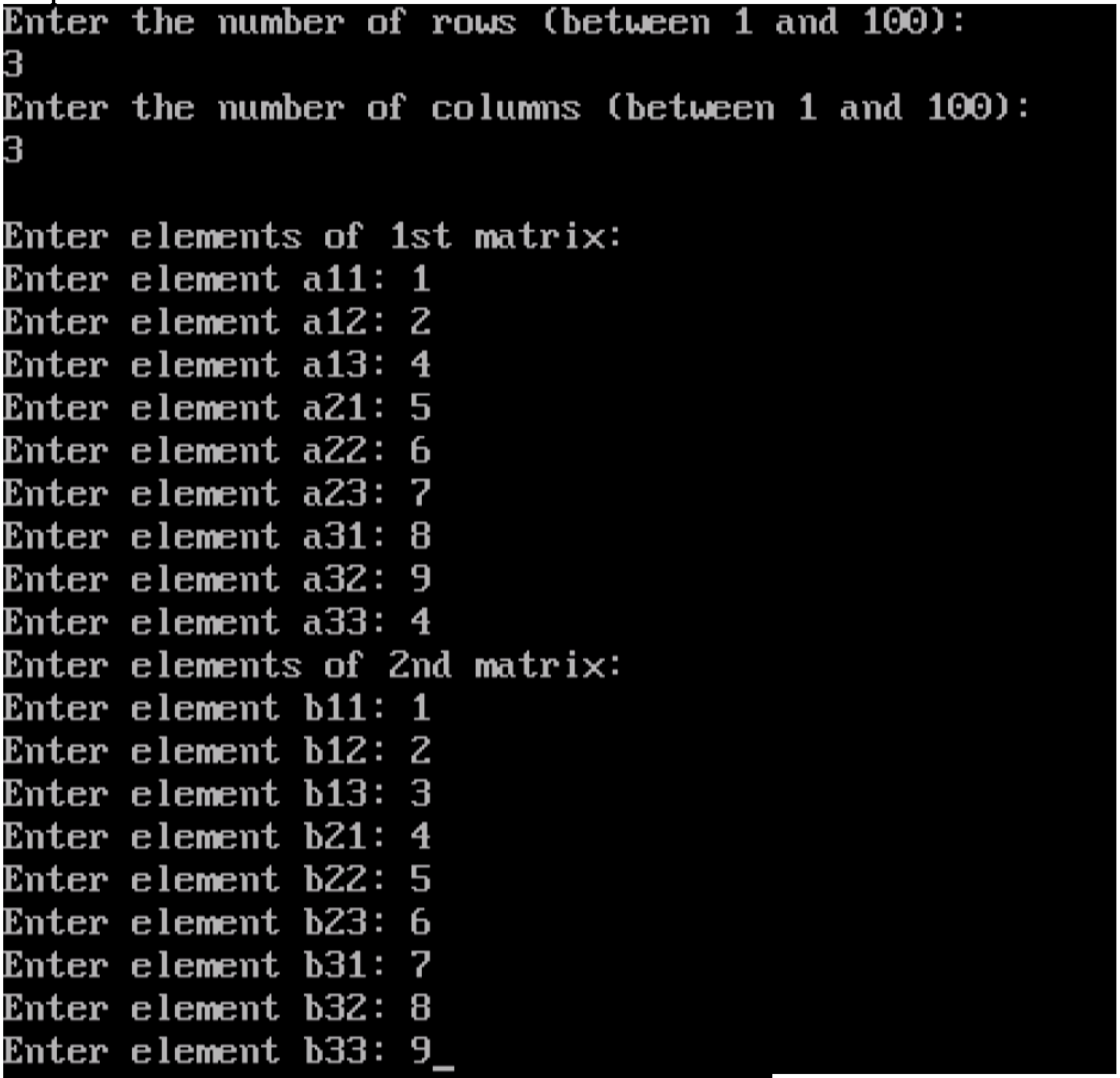
Algorithm:

1. Start the program.
2. Declare the variables.
3. Get the Input as 2 array elements from user. Store it in a variable.
4. Add the two matrix value and store it
5. Print the result.
6. Stop the program.

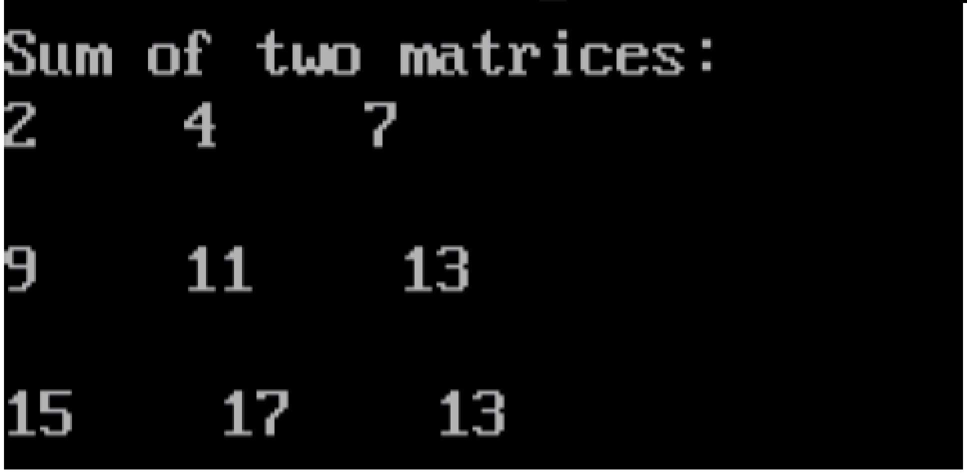
Program:

```
 #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]);
    }

  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j)
    {
      sum[i][j] = a[i][j] + b[i][j];
    }
  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: 4
Enter element a21: 5
Enter element a22: 6
Enter element a23: 7
Enter element a31: 8
Enter element a32: 9
Enter element a33: 4
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     7


9     11     13


15     17     13
```

Result:

Thus the program was compiled and executed successfully

EX.NO:8(ii)

Date:

Multiplication two matrices

Aim:

To write C Program to perform Multiplication of two matrices.

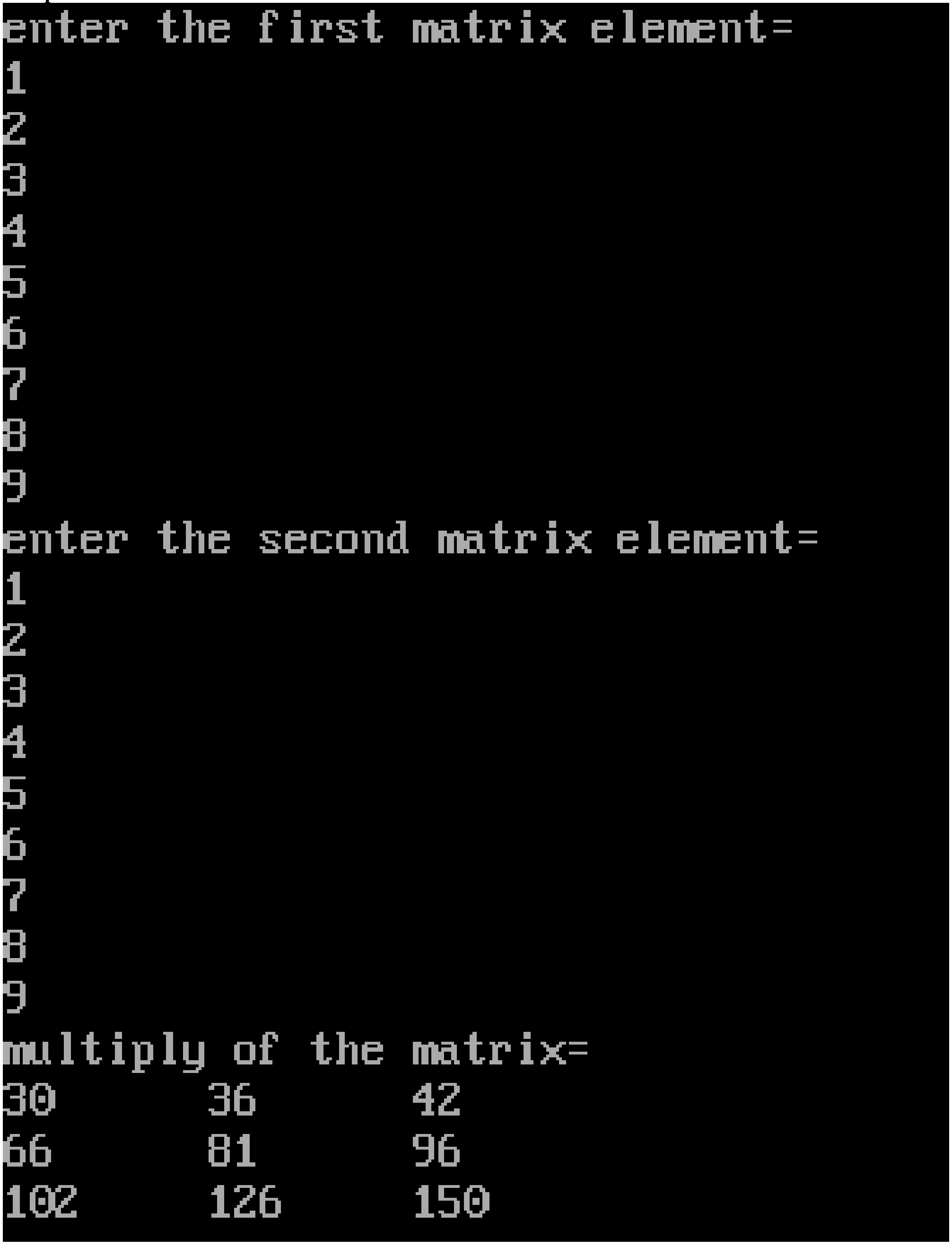
Algorithm:

1. Start the program.
2. Declare the variables.
3. Get the Input as 2 array elements from user. Store it in a variable.
4. Multiplication the two matrix value and store it
5. Print the result.
6. Stop the program.

Program:

```
  #include<stdio.h>
#include<stdlib.h>
int main(){
int a[10][10],b[10][10],mul[10][10],r,c,i,j,k;
system("cls");
printf("enter the number of row=");
scanf("%d",&r);
printf("enter the number of column=");
scanf("%d",&c);
printf("enter the first matrix element=\n");
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
scanf("%d",&a[i][j]);
}
}
printf("enter the second matrix element=\n");
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
scanf("%d",&b[i][j]);
}
}

printf("multiply of the matrix=\n");
for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
mul[i][j]=0;
for(k=0;k<c;k++)
{
mul[i][j]+=a[i][k]*b[k][j];
}
}
}

for(i=0;i<r;i++)
{
for(j=0;j<c;j++)
{
printf("%d\t",mul[i][j]);
}
printf("\n");
}
return 0;
}
```

Output:



```
enter the first matrix element=
1
2
3
4
5
6
7
8
9
enter the second matrix element=
1
2
3
4
5
6
7
8
9
multiply of the matrix=
30       36       42
66       81       96
102      126      150
```

Result:

Thus the program was compiled and executed successfully

EX.NO:9(iii)	
Date:	Arithmetic Operations Using Pointer

Aim:

To write C Program to demonstrate the arithmetic operations using pointers.

Algorithm:

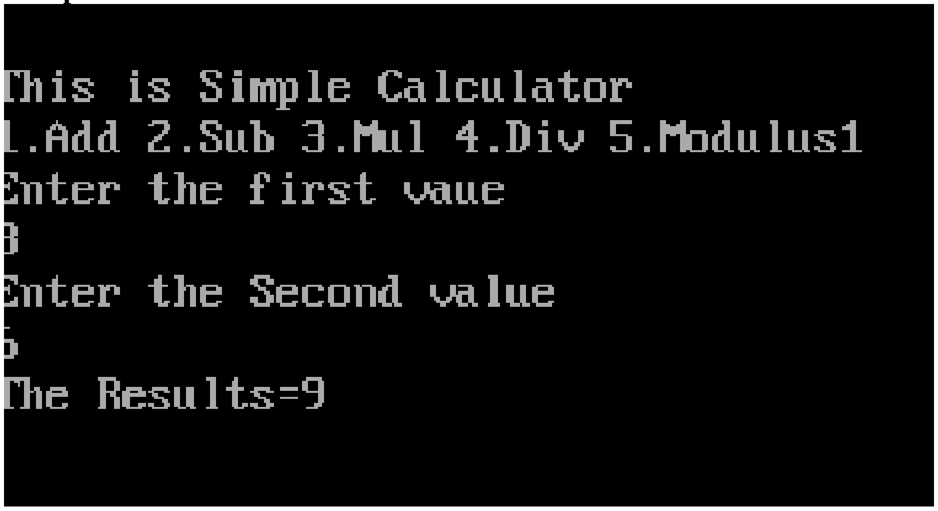
1. Start the program.
2. Create a pointer variable, which points to an integer data.
3. Get the input from the user like choice, first and second value.
4. The program perform the arithmetic operation based on the choice, first and second value, Store the result
5. Pointers are used to store Address of variables or a memory location. In C, a pointer is created by placing a * before the variable name. We can access the value of the pointer by using * before the name of the pointer variable.

   For example, **int *x;** /* this creates a pointer with name x */
6. Print the result.
7. Stop the program.

Program:

```
#include
<stdio.h>
#include<conio.h
>
  int main()
 {
   int a,b,ch;
   int *pt1,*pt2;
   clrscr();
   printf("This is Simple Calculator");
   printf("\n1.Add 2.Sub 3.Mul 4.Div
   5.Modulus"); scanf("%d",&ch);
   printf("Enter the first vaue");
   scanf("\n%d",&a);
   printf("Enter the Second value");
   scanf("\n%d",&b);
   pt1=&a;
   pt2=&b;
   switch(ch)
  {
   case 1:
    printf("The Results=%d",(*pt1 +
   *pt2)); break;
   case 2 :
    printf("The Results=%d",(*pt1 -
   *pt2)); break;
   case 3:
    printf("The Results=%d",(*pt1 *
   *pt2)); break;
   case 4:
    printf("The Results=%d",(*pt1 /
   *pt2)); break;
   case 5:
    printf("The Results=%d",(*pt1 % *pt2));
   break;
  }
  return 0;
}
```

Output:



```
This is Simple Calculator
1.Add 2.Sub 3.Mul 4.Div 5.Modulus1
Enter the first vaue
3
Enter the Second value
6
The Results=9
```

Result:

Thus the program was compiled and executed successfully

EX.NO:10(i)

Date:

Structures and Unions

Aim:

To Write a C program to demonstrate the differences between structures and unions.

Algorithm:

1. Start the program.
2. Create a structure and union.
3. Structures in C are a user-defined data type available in C that allows to combining of data items of different kinds. Structures are used to represent a record.

```
struct [structure name]
  {
     member definition;
     member definition;
     ...
     member definition;
  }structure variable declaration;
```

4. Union in C is a special data type available in C that allows storing different data types in the same memory location. You can define a union with many members, but only one member can contain a value at any given time. Unions provide an efficient way of using the same memory location for multiple purposes.

```
union [union name]
  {
    member definition;
    member definition;
    ...
    member definition;
  }union variable declaration;
```

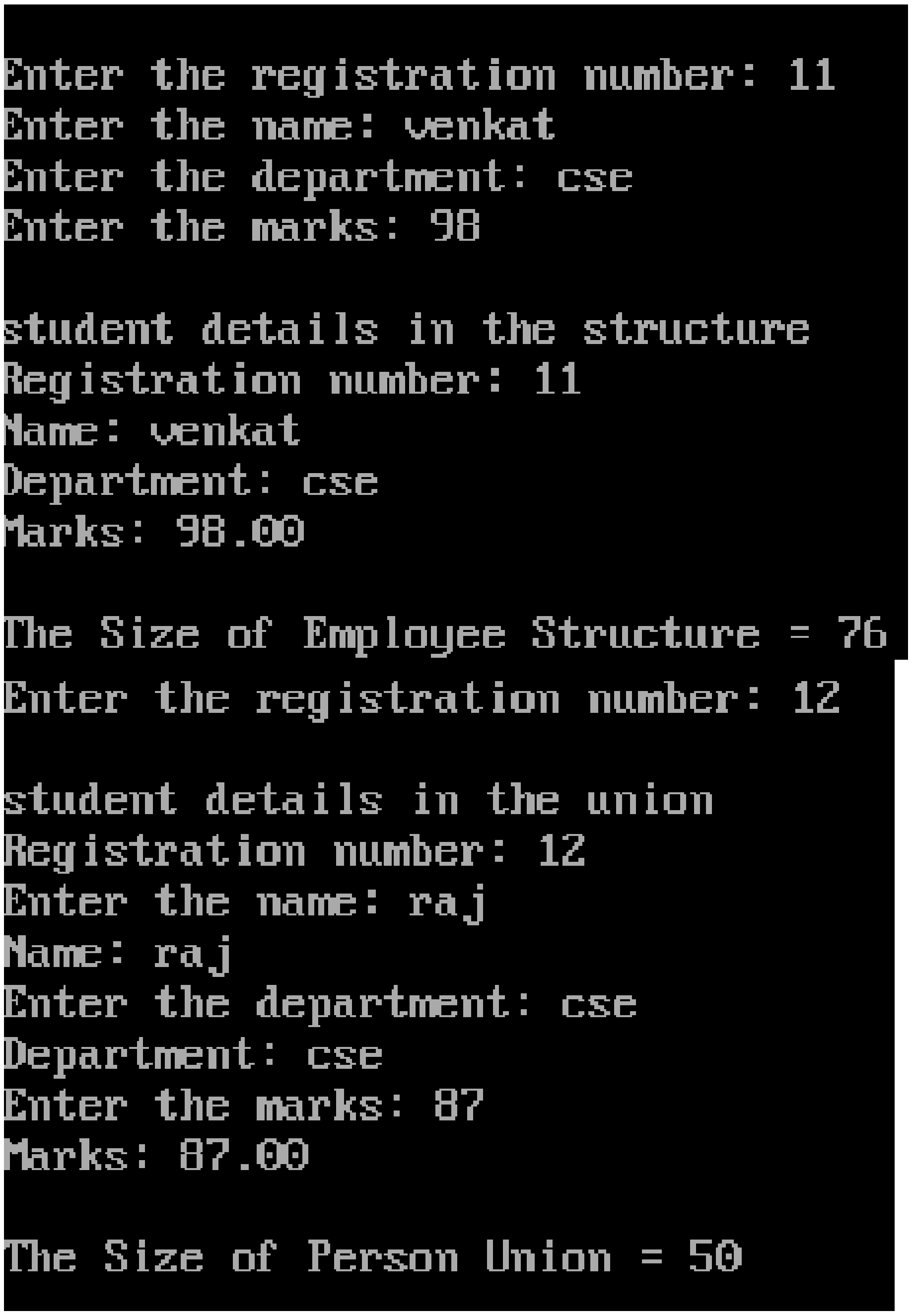
5. The main difference of structure Each variable member occupied a unique memory space, Each variable member will be assessed at a time and union Variables members share the memory space of the largest size variable and Only one variable member will be assessed at a time.
6. Get the input value from user.
7. Print the result and the size of union and structure.
8. Stop the program.

Program:

```
#include <stdio.h>
struct Employee
{
  int regno;
  char Name[50];
  char Department[20];
  float marks;
};
union Person
{
   int regno;
  char Name[50];
  char Department[20];
  float marks;
};
int main()
{
  struct Employee e1;
  union Person P1;
  printf("Enter the registration number: ");
  scanf("%d", &e1.regno);
  printf("Enter the name: ");
  scanf("%s", e1.Name);
  printf("Enter the department: ");
  scanf("%s", e1.Department);
  printf("Enter the marks: ");
  scanf("%f", &e1.marks);
  printf("student details in the structure");
  printf("\nRegistration number: %d\n", e1.regno);
  printf("Name: %s\n", e1.Name);
  printf("Department: %s\n", e1.Department);
  printf("Marks: %.2f\n", e1.marks);
  printf(" The Size of Employee Structure = %d\n", sizeof (e1) );
  printf("Enter the registration number: ");
  scanf("%d", &p1.regno);
  printf("Enter the name: ");
  scanf("%s", p1.Name);
  printf("Enter the department: ");
  scanf("%s", p1.Department);
```

```
 printf("Enter the marks: ");
 scanf("%f", &p1.marks);
 printf("student details in the union");
 printf("\nRegistration number: %d\n", p1.regno);
 printf("Name: %s\n", p1.Name);
 printf("Department: %s\n", p1.Department);
 printf("Marks: %.2f\n", p1.marks);
 printf(" The Size of Person Union = %d\n", sizeof (P1));
 return 0;
}
```

Output:



```
Enter the registration number: 11
Enter the name: venkat
Enter the department: cse
Enter the marks: 98

student details in the structure
Registration number: 11
Name: venkat
Department: cse
Marks: 98.00

The Size of Employee Structure = 76
Enter the registration number: 12

student details in the union
Registration number: 12
Enter the name: raj
Name: raj
Enter the department: cse
Department: cse
Enter the marks: 87
Marks: 87.00

The Size of Person Union = 50
```

Result:

Thus the program was compiled and executed successfully

EX.NO:11(ii)	Find the Length of a String
Date:	

Aim:

To write a C function to find the length of a string.

Algorithm:

1. Start the program.
2. Declare the variable.
3. Get the values from the user.
4. The **call by value** method of passing arguments to a function copies the actual value of an argument into the formal parameter of the function. In this case, changes made to the parameter inside the function have no effect on the argument.
5. Calculate the string length without using built in function using for loop.
6. Calculate the string length with using built in function Strlen().
7. Store the result and Print the result.
8. Stop the program.

Program:

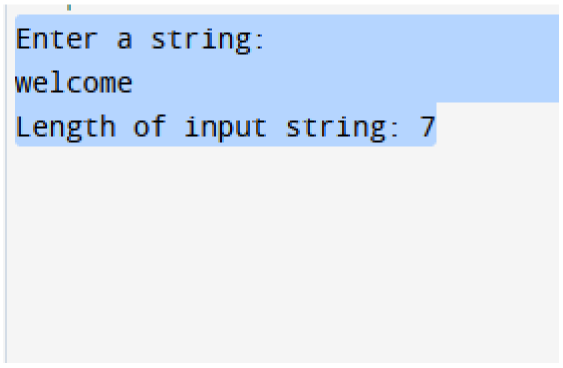
(i)// calculate the string length without using built in function

```
#include
<stdio.h> int
main()
{
   char
   str[100]; int
   i,length=0;
   printf("Enter a string: \n");
   scanf("%s",str);
   for(i=0; str[i]!='\0'; i++)
   {
      length++; //Counting the length.
   }

   printf("\nLength of input string: %d",length);

    return 0;
}
```

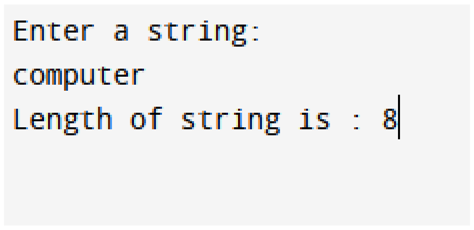
(ii)// calculate the string length with using built in function

```
#include
<stdio.h> int
main()
{
   char
   str[100]; int
   length=0;
   printf("Enter a string: \n");
   scanf("%s",str);
  length = strlen(str);
    printf("Length of string is : %d", length);

    return 0;
 }
```

Output:



```
Enter a string:
welcome
Length of input string: 7
```



```
Enter a string:
computer
Length of string is : 8
```

Result:

Thus the program was compiled and executed successfully.

EX.NO:9(iii)	
Date:	Factorial of a number using recursive function

Aim:

To write C Program to find the factorial of a number using recursive function.

Algorithm:

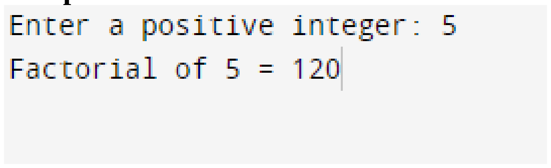
1. Start the program.
2. The multiplication of all positive integers, say "n", that will be smaller than or equivalent to n is known as the factorial. The factorial of a positive integer is represented by the symbol "**n!**".
3. The formula to find the factorial of a number is

   n! = n × (n-1) × (n-2) × (n-3) × ….× 3 × 2 × 1 For an integer $n \geq 1$.
4. Recursion is the process of a function calling itself repeatedly till the given condition is satisfied. A function that calls itself directly or indirectly is called a recursive function and such kind of function calls are called recursive calls.
5. Create a recursive function to find the GCD of two numbers.
6. Create a variable and get the user input.
7. To find the factorial of a number.
8. Print Result.
9. Stop the program.

Program:

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

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

Output:



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

Result:

Thus the program was compiled and executed successfully

EX.NO:13(i)	Swap Two Numbers Using Call By Reference
Date:	

Aim:

To write a C program to swap two numbers using call by reference.

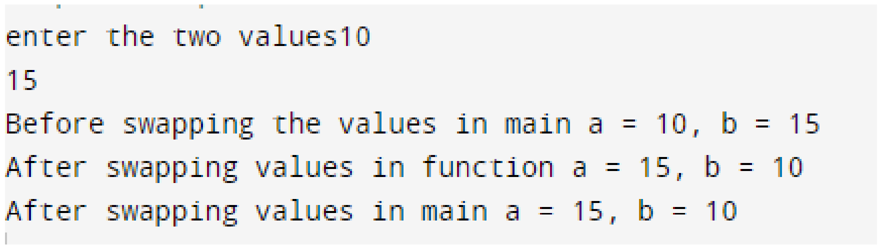
Algorithm:

1. Start the program.
2. In call by reference, the address of the variable is passed into the function call as the actual parameter. The value of the actual parameters can be modified by changing the formal parameters since the address of the actual parameters is passed.
3. In call by reference, the memory allocation is similar for both formal parameters and actual parameters. All the operations in the function are performed on the value stored at the address of the actual parameters, and the modified value gets stored at the same address.
4. Create a e function to swap two numbers.
5. Create a variable and get the user input.
6. To generate Fibonacci series.
7. Print Result.
8. Stop the program.

Program:

```
#include <stdio.h>
void swap(int *, int *); //prototype of the function
int main()
{
    int a,b;
    printf("enter the two values");
    scanf("%d%d",&a,&b);
    printf("Before swapping the values in main a = %d, b = %d\n",a,b); // printing
the value of a and b in main
    swap(&a,&b);
    printf("After swapping values in main a = %d, b = %d\n",a,b); // The values of
actual parameters do change in call by reference, a = 10, b = 20
}
void swap (int *a, int *b)
{
    int temp;
    temp = *a;
    *a=*b;
    *b=temp;
    printf("After swapping values in function a = %d, b = %d\n",*a,*b); // Formal
parameters, a = 20, b = 10
}
```

Output:



```
enter the two values10
15
Before swapping the values in main a = 10, b = 15
After swapping values in function a = 15, b = 10
After swapping values in main a = 15, b = 10
```

Result:

Thus the program was compiled and executed successfully

EX.NO:13(ii)

Date:

Copy One String into another Using Pointer

Aim:

To write a C program to copy one string into another using pointer.

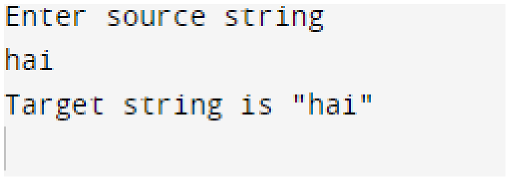
Algorithm:

1. Start the program.
2. In call by reference, the address of the variable is passed into the function call as the actual parameter. The value of the actual parameters can be modified by changing the formal parameters since the address of the actual parameters is passed.
3. In call by reference, the memory allocation is similar for both formal parameters and actual parameters. All the operations in the function are performed on the value stored at the address of the actual parameters, and the modified value gets stored at the same address.
4. Create a variable and get the user input.
5. Create a function to copy one string into another using pointer.
6. Print Result.
7. Stop the program.

Program:

```
#include<stdio.h>
void copy_string(char*,
char*); main()
{
    char source[100],
    target[100]; printf("Enter
    source string\n");
    gets(source);
    copy_string(target, source);
    printf("Target string is \"%s\"\n", target);
    return 0;
}

void copy_string(char *target, char *source)
{
    while(*source)
    {
        *target =
        *source;
        source++;
        target++;
    }
    *target = '\0';
}
```

Output:



```
Enter source string
hai
Target string is "hai"
```

Result:

Thus the program was compiled and executed successfully

EX.NO:14(i)

Date:

Write and Read Text into a File

Aim:

To write a C program to write and read text into a file.

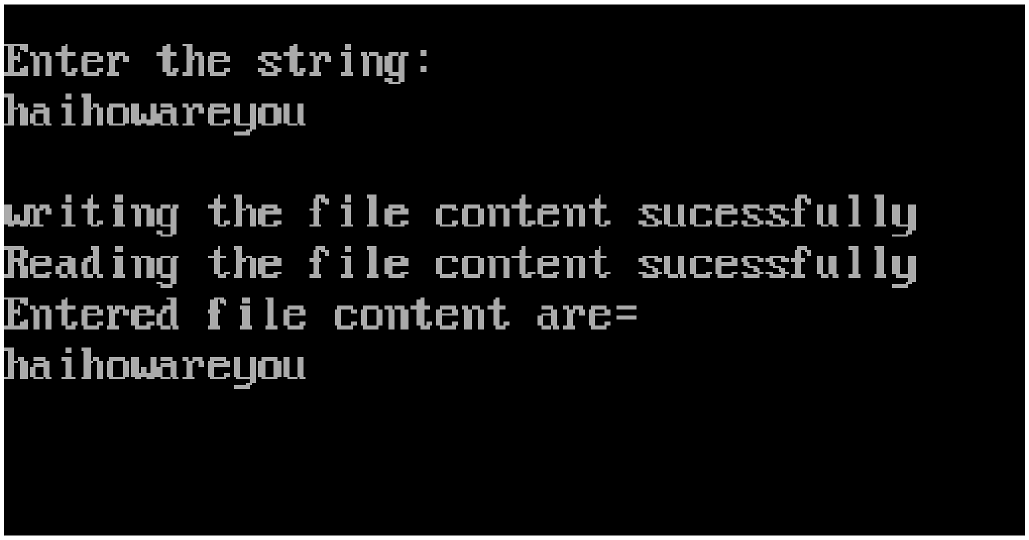
Algorithm:

1. Start the program.
2. Creating variable of file pointer
3. Opening/creating a file
4. Write some of the characters in a file
5. Closing a file
6. Reading one by one character from a file, Reading all text until EOF (End of the file) is not found.
7. Closing the file
8. Print Result.
9. Stop the program.

Program:

```
#include
<stdio.h>
#include<conio.h
>           #include
<stdlib.h>
  int main()
   {
    char
    str[100];
    FILE *fptr;
    clrscr();
    fptr = fopen("C:\program.txt","w");

    if(fptr == NULL)
     {
      printf("Error!");
      exit(1);
     }
    printf("Enter the string: ");
    scanf("%s",str);
    fprintf(fptr,"%s",str);
    printf("\nwriting the file content sucessfully");
    fclose(fptr);

    if ((fptr = fopen("C:\program.txt","r")) == NULL)
     {
      printf("Error! opening
      file"); exit(1);
     }

  fscanf(fptr,"%s",&str);
  printf("\nReading the file content sucessfully");
  printf("\nEntered file content are=\n%s",str);
  fclose(fptr);
    return 0;
  }
```

Output:



```
Enter the string:
haihowareyou

writing the file content sucessfully
Reading the file content sucessfully
Entered file content are=
haihowareyou
```

Result:

Thus the program was compiled and executed successfully

EX.NO:14(ii)

Date:

Write and Read Text into a Binary File

Aim:

To write a C program to write and read text into a binary file using fread() and fwrite().

Algorithm:

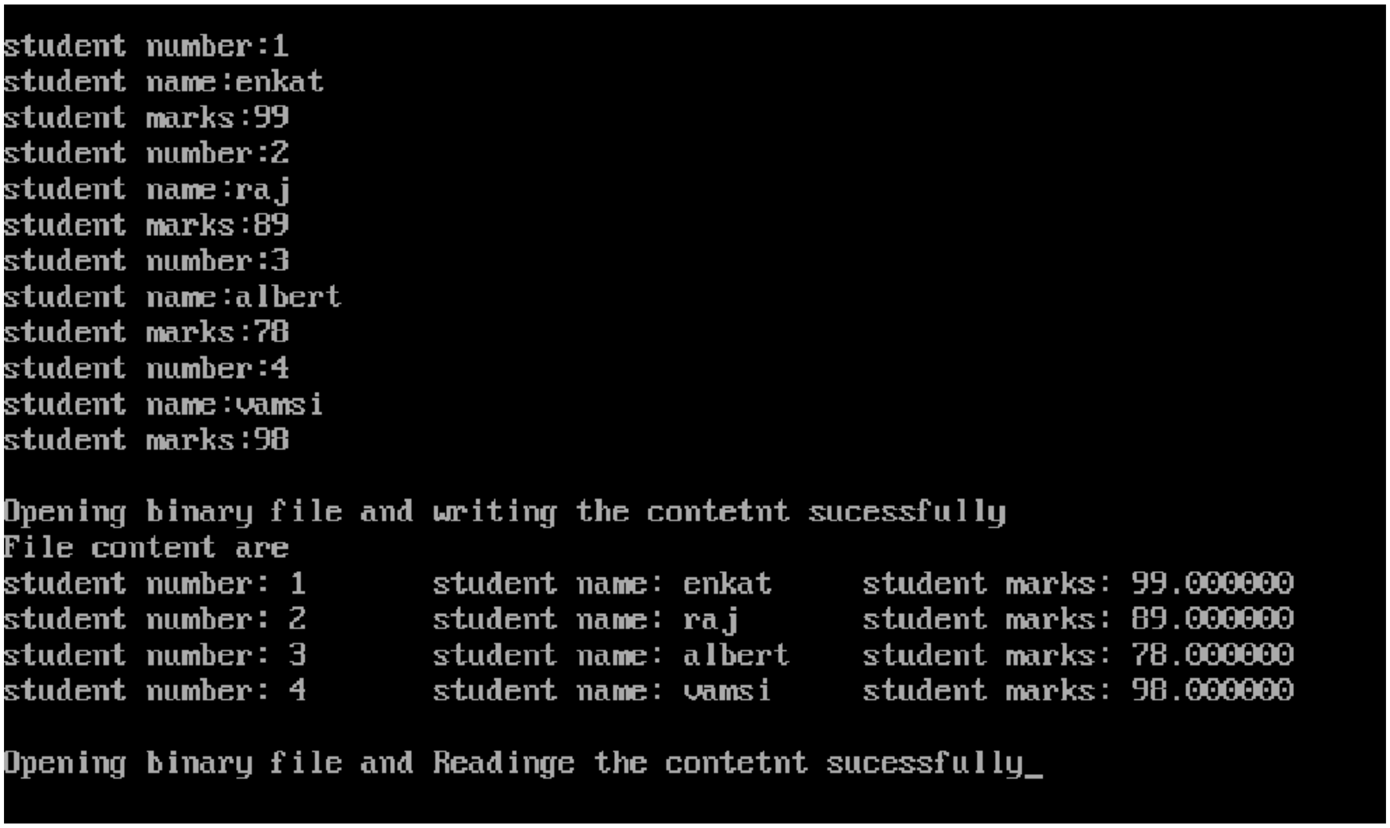
1. Start the program.
2. Creating variable of file pointer
3. Create/open in write mode a Binary file with name “program.bin”.
4. written the file in a binary way by using the function fwrite
5. fwrite(addressData, sizeData, numbersData, pointerToFile);
6. Reading the character from a file
7. Closing a file.
8. Reopen the file for reading it.
9. Read the file by using the function fread, Reading all text until EOF (End of the file) is not found.
10. fread(addressData, sizeData, numbersData, pointerToFile);
11. closing the file
12. Print Result.
13. Stop the program.

Program:

```
#include
<stdio.h>
#include<conio.h
>           #include
<stdlib.h>   struct
student{
   int sno;
   char sname
   [30]; float
   marks;  char
   temp;
};
int main()
{
   int n;
   struct student s;
   FILE *fptr;
   clrscr();
   if ((fptr = fopen("C:\\program.bin","wb")) == NULL){
       printf("Error! opening file");
       exit(1);
   }
   for(n = 1; n < 5; ++n)
   {
      printf("student
      number:");
      scanf("%d",&s.sno);
      scanf("%c",&s.temp);
      printf("student name:");
      gets(s.sname);
      printf("student marks:");
      scanf("%f",&s.marks);
      fwrite(&s, sizeof(struct student), 1, fptr);
       }
   fclose(fptr);
   printf("\nOpening binary file and writing the contetnt sucessfully");

   if ((fptr = fopen("C:\\program.bin","rb")) == NULL){
       printf("Error! opening file");
```

```
            exit(1);
          }
          printf("\nFile content are\n");

for(n = 1; n < 5; ++n)
          {
            fread(&s, sizeof(struct student), 1, fptr);
            printf("student number: %d\tstudent name: %s\tstudent marks: %f\n", s.sno, s.sname,
            s.marks);
          }
           printf("\nOpening binary file and Readinge the contetnt
           sucessfully"); fclose(fptr);
           return 0;
        }
```

Output:



```
student number:1
student name:enkat
student marks:99
student number:2
student name:raj
student marks:89
student number:3
student name:albert
student marks:78
student number:4
student name:vamsi
student marks:98

Opening binary file and writing the contetnt sucessfully
File content are
student number: 1       student name: enkat     student marks: 99.000000
student number: 2       student name: raj       student marks: 89.000000
student number: 3       student name: albert    student marks: 78.000000
student number: 4       student name: vamsi     student marks: 98.000000

Opening binary file and Readinge the contetnt sucessfully_
```

Result:

Thus the program was compiled and executed successfully