

Laboratory Journal
of
NUMERICAL METHODS IN COMPUTER
PROGRAMMING

*For completion of term work of 4th semester
curriculum program*

Bachelor of Technology
In
ELECTRONICS AND TELECOMMUNICATION ENGINEERING



DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION
ENGINEERING

Dr. BABASAHEB AMBEDKAR TECHNOLOGICAL UNIVERSITY

Lonere-402 103, Tal. Mangaon, Dist. Raigad (MS)

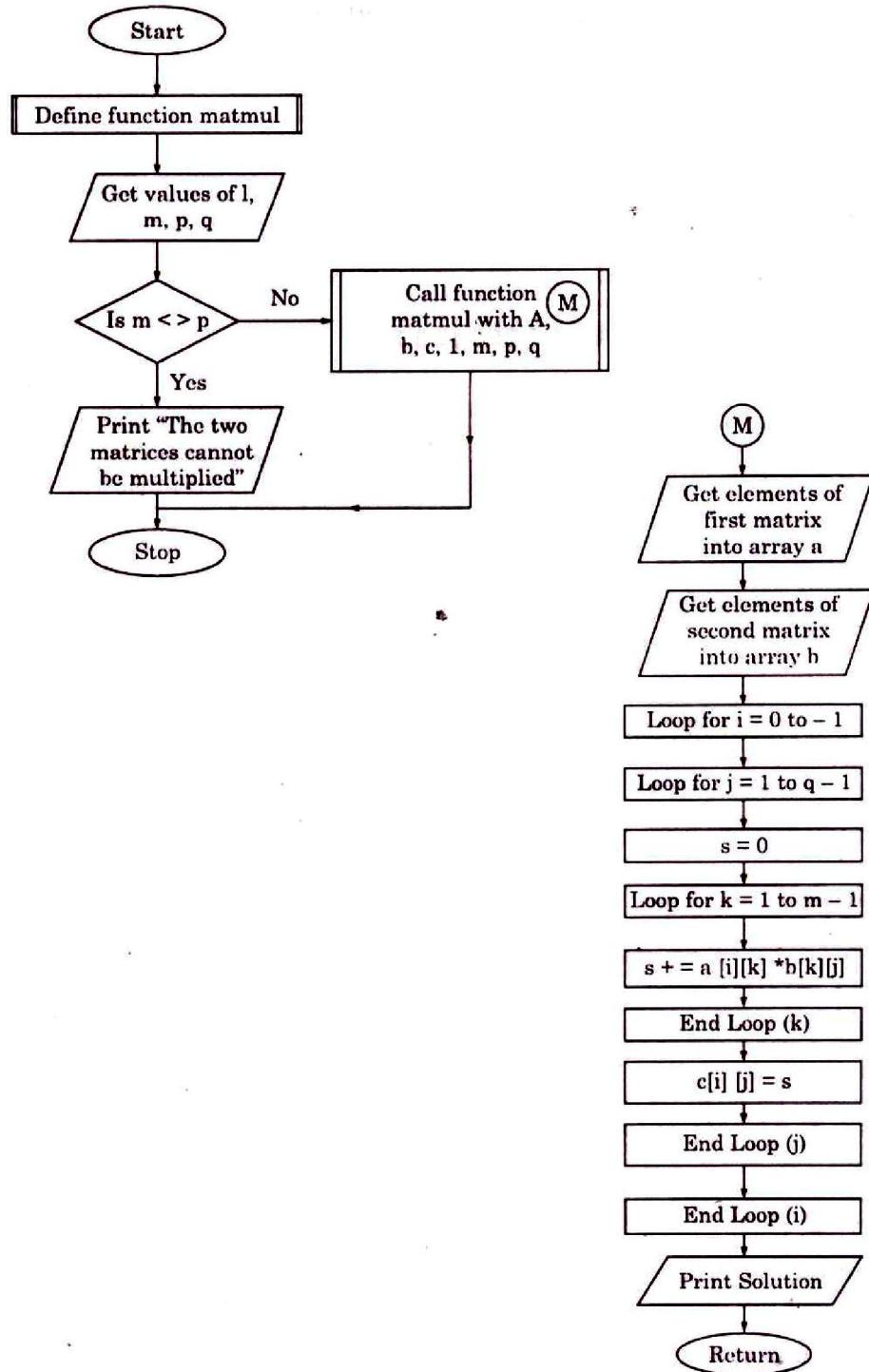
INDIA

Exp. No.	Title
1	Flow chart & C program of Multiplication of Matrices
2	Flow chart & C program of Bisection Method
3	Flow chart & C program of Newton Ramphson Method
4	Flow chart & C program of Gauss Elimination Method
5	Flow chart & C program of Gauss Jordan method
6	Flow chart & C program of Euler's Method
7	Flow chart & C program of Runga-Kutta Method
8	Flow chart & C program of Trapezoidal Rule
9	Flow chart & C program of Simpson's 1/3rd Rule
10	Flow chart & C program of Newton Forward Method
11	Flow chart & C program of Modified Euler's Method
12	Flow chart & C program of Least Square Method
13	Flow chart & C program of Modified Euler's Method
12	Flow chart & C program of Least Square Method

Experiment No: 1

AIM : Flow chart & C program of Multiplication of Matrices

A) Flow chart – Program for Multiplication of Matrices



B) C- Program

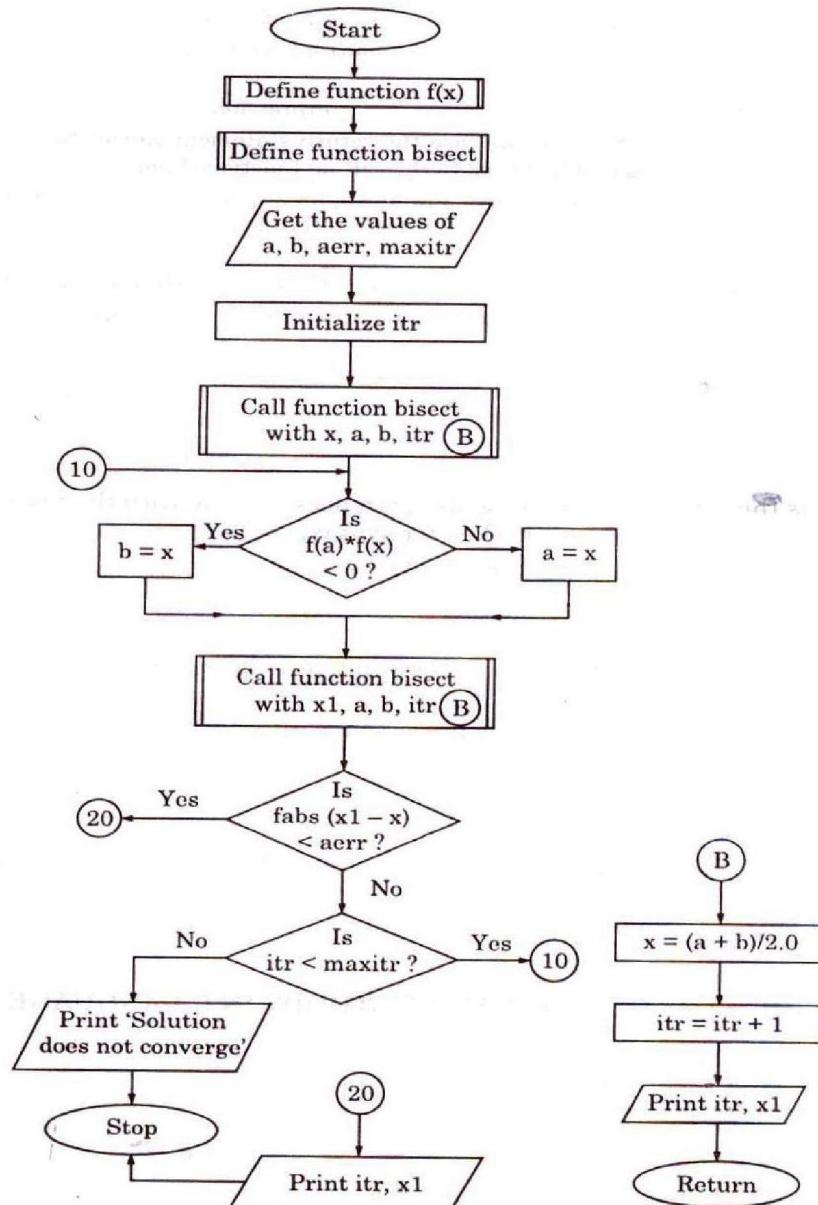
```
# include < stdio. h>
# include < conio. h>
# include < math. h>
main ( )
{
int a[3][3],b[3][3], c[3][3];
int i, j, k
clrscr ();
printf ("MULTIPLICATION OF MATRICES");
printf ("ENTER FIRST MATRIX \n");
for (i=0;i<3;i++)
{
for (j=0;j<3;j++)
{
scanf ("%d",& a[i][j]);
}
}
printf("Enter second matrix");
for (i=0;i<3;i++)
{
for (j=0;j<3;j++)
{
Scanf("%d",& b[i][j]);
}
}
for (i=0;i<3;i++)
for (j=0;j<3;j++)
c[i][j]=0;
printf ("Multiplication of Two Matrix");
for (i=0;i<3;i++) for
(j=0;j<3;j++) c[i][j]=c[i][j]+(
a[i][k]* b[k][j]; for
(i=0;i<3;i++)
{
printf ("\n");
for (j=0;j<3;j++)
printf ("%d\t", c[i][j]);
}
}
getch();
}
```

Experiment No: 2

AIM

: Flow chart & C program of Bisection Method

A) Flow chart



B) C- Program

```
/* Bisection Method */
#include < stdio.h>
#include < math.h>
float y(float x)
{
    return (x*x*x - 4*x-9);
}
Void bisect (float *x, float a, float b, int* int)
(*x = (a +b)/2;
++(*itr);
printf("Iteration no. % 3d x = % .7.5 f \n",*x);
}
main ()
{
int int=0, maxitr;
float x,a,b,aerr,x1;
printf("Enter the value of a,b" "allowed error , maximum iterations \n");
scanf ("%f %f %f %d, & a ,& b ,& aerr, &maxitr );
bisect (&x, a, b, &itr);
do
{
if (f(a)*f(x)<0)
b=x;
else
a=x;
bisect (&x1, a, b, &itr);
if ( fabs (x1-x)< aerr )
{
printf ("After % d iteration , root" =% .6.4\n",itr,x1);
return 0;
}
x=x1;
}
While (itr < maxtr ) ;
printf ("Solution does not converge," "iterations not sufficient ");
return 1;
}
```

Output -

Enter the value of a, b” “allowed error, maximum iterations

3 2 0.0001 20

Iteration No. 1 x = 2.50000

Iteration No. 1 x = 2.75000

Iteration No. 1 x = 2.625000

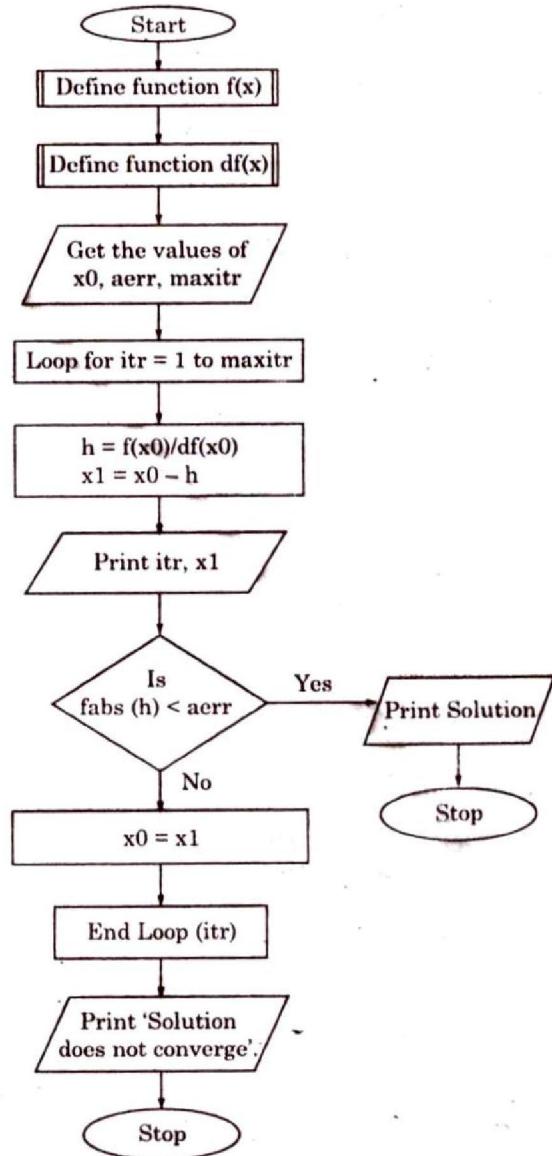
Iteration No. 1 x = 2.68750

Experiment No: 3

AIM

: Flow chart & C program of Newton Ramphson Method

A) Flow chart: Flow chart of Newton Raphson Method



B) C- Program

```
/* Newton Ramphson Method */
#include < stdio.h>
#include < math.h>
float f (float x)
{
    return x*log 10(x)-1.2;
}
float dx( float x);
{
    return log 10(x)+0.43429 ;
}
main ( )
{
    int int, maxitr;
    float h,x0,x1,aerr;
    printf("Enter the value of x0, allowed error , maximum iterations \n");
    scanf( "%f %f %d", &x0 ,&aerr, &maxitr );
    for (itr=1; itr,=maxitr; itr++)
    {
        h=f(x0)/ df(x0);
        x1=x0-h;
        printf("Iteration no. %3d,"
               "x= %9.6f\n", itr, x1);
        if (fabs (h) < aerr)
        {
            printf("After % 3d iterations," "root = % 8.6 f\n, itr,x1);
            return 0;
        }
        x0=x1;
    }
    printf("Iterations not sufficient ,," " solution does not converge\n");
    return 1;
}
```

Enter the value of x0, allowed error, maximum iterations

2 0.000001 10

Iteration No. 1 x = 2.813170

Iteration No. 2 x = 2.741109

Iteration No. 3 x = 2.740646

Iteration No. 4 x = 2.740646

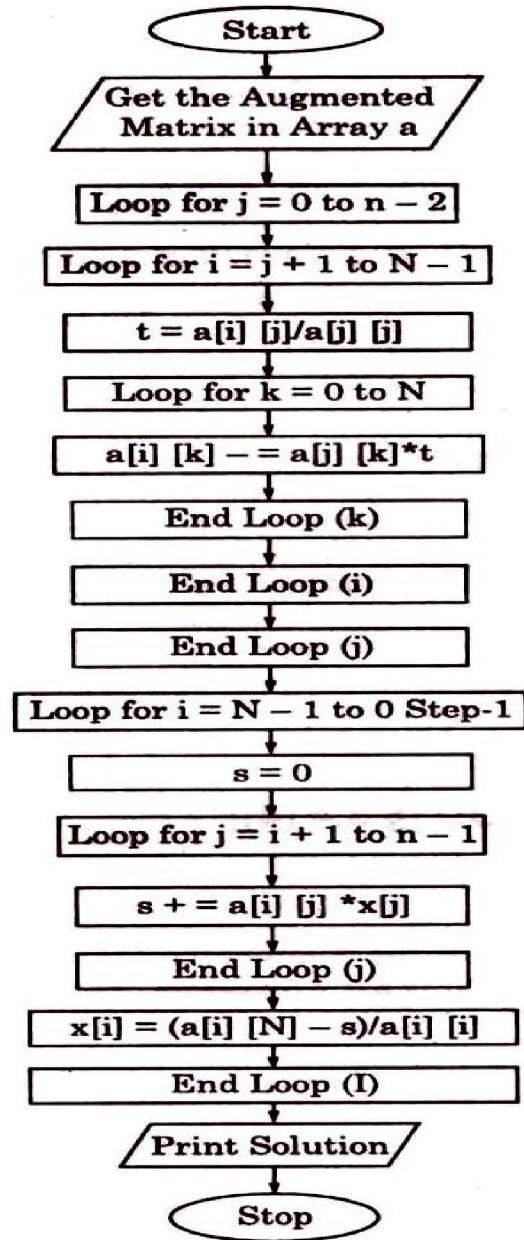
After 4 Iterations, root =2.740646

Experiment No: 4

AIM

: Flow chart & C program of Gauss Elimination Method

A) Flow chart



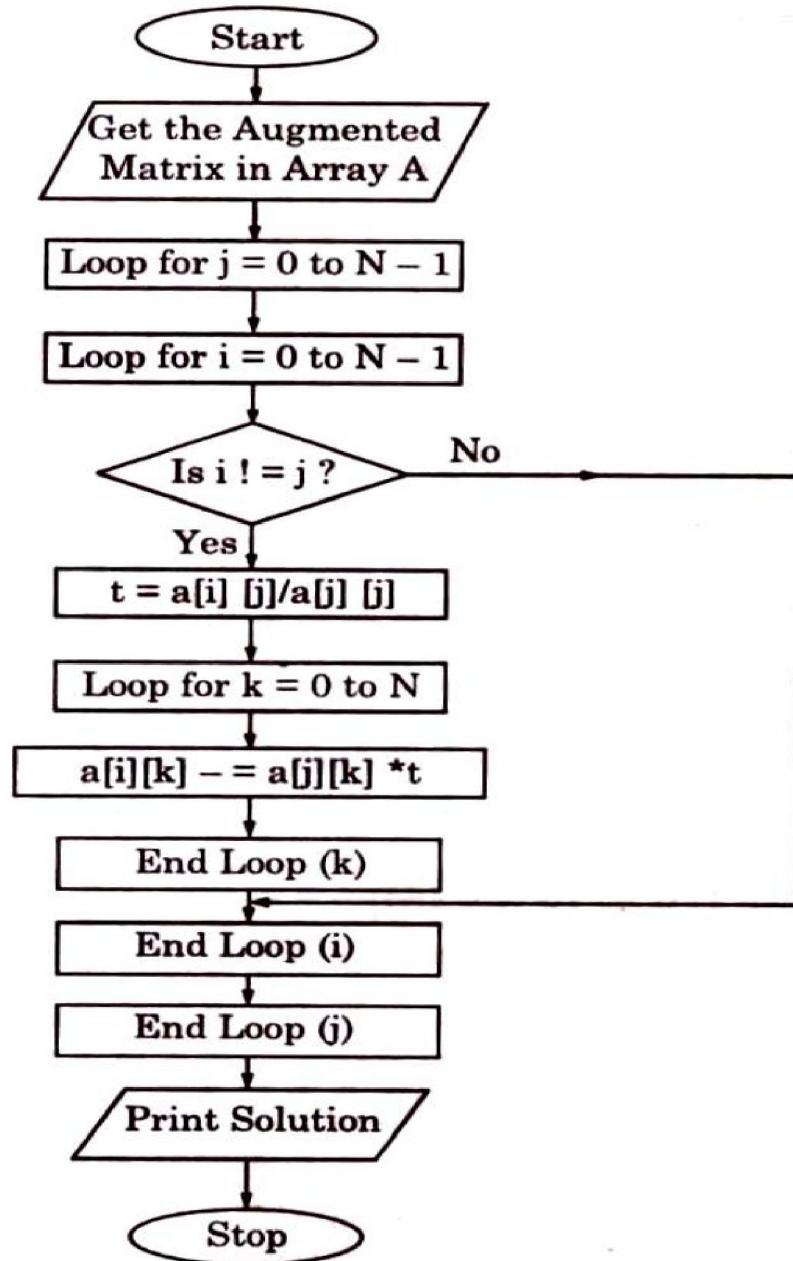
B) C- Program

```
/* Gauss elimination method */
# include <studio.h>
#define N 4
main()
{
    float a[N][N+1], x[N], t, s;
    int i, j, k;
    print f("enter the elements of the "
           "augmented matrix row wise \n");
    for( i=0 ; i<N; i++ )
        for (j=0; j<N+1 ; j++)
            Scan f("%f", &a [i] , [j] );
    for (j=0; j<N-1; j++)
        for (i=j+1; i<N ; i++)
    {
        t= a[i] [j] /a[j] [j];
        for (k=0; k<N+1; k++)
            a [i] [k]=a [j] [k] *t;
    }
    /* now printing the
       upper triangular matrix */
    print f("The upper triangular matrix "
           "is :- \n")
    for ( i=0 ; i<N ;i++)
    {
        for (j=0; j<N+1 ; j++)
            print f("%8 . 4f", a [i] [j] );
        print f("\n");
    }
    /* now performing back substitution */
    for (i=N-1 ; i>=0 ; i- -)
    {
        s= 0 ;
        for (j=i+1 ; j<N; j++)
            s += a[i] [j] *x[j] ;
        x[i] = (a[i] [N-s ])/a [i] [i] ;
    }
    /* now printing the results */
    print f(" The solution is :- \n");
    for (i=0 ; i<N; i++)
        print f("x[%3d] = %7 . 4f\n", i+1, x[i]);
}
```

Experiment No: 5

AIM : Flow chart & C program of Gauss Jordan method

A) Flow chart



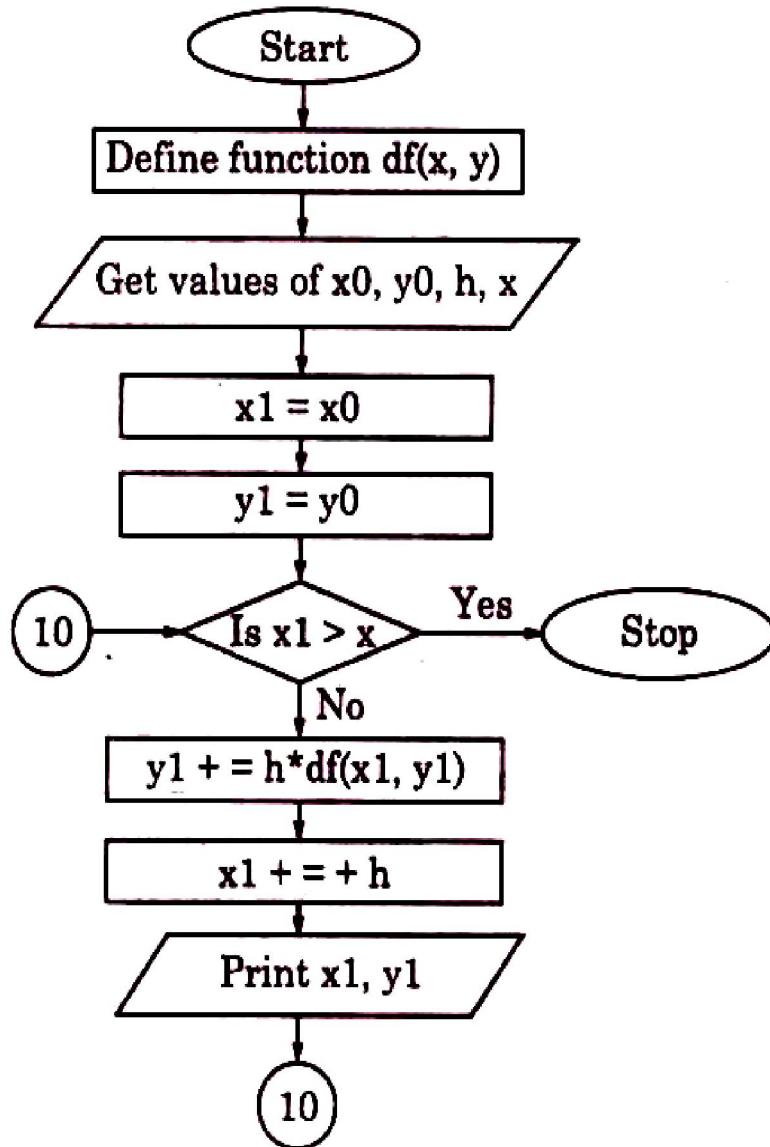
B) C- Program

```
/* Gauss Jordan method */
#include <stdio.h>
#define N 3
main ( )
{
float a [N] [N+1] , t;
int i, j, k;
print f ("Enter the elements of the "
        "augmented matrix row wise\n")
for (i=0; i<N; i++)
    for (j=0 ;j<N+1;j++)
        scan f ("%f ", &a[i] [j]);
/* now calculating the values
   of x1,x2, ....,xN */
for (j=0; j<N; j++)
    for (i=0; i<N;i++)
if (i!= j)
{
    t= a [i] [j];
    for ( k=0 ;k<N+1; k++)
        a[i] [k] -= a[j] [k]*t;
}
/* now printing the diagonal matrix */
print f ("The diagonal matrix is :- ,\n");
{
for (i= 0; i<N; i++)
    for (j=0 ;j<N+1; j++)
        print f ("%9. 4f ", a[i] [j]);
    print f ("\n");
}
/* now printing the results */
print f ("The solution is :- \n");
for (i=0; i<N ;i++)
print f ("x[%d] = %7.4f\n",
        i+1, a[i] [N]/a[i] [i]);
}
```

Experiment No: 6

AIM : Flow chart & C program of Euler's Method

A) Flow chart.



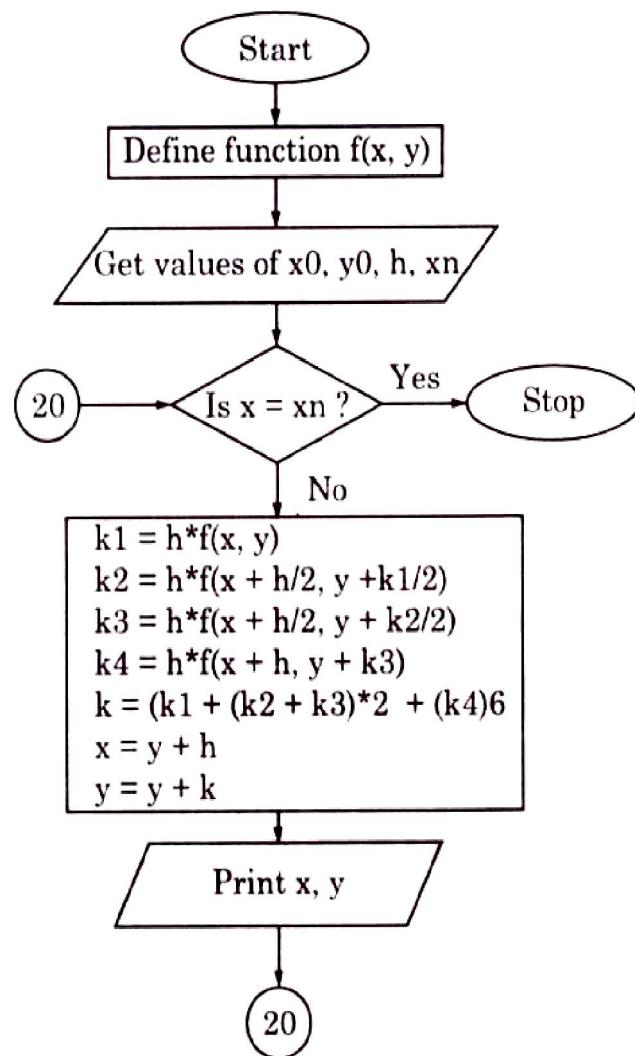
B) C- Program

```
/* Euler's Method */  
# include < stdio.h>  
float df(float x, float y)  
{  
    return x+y;  
}  
main ()  
{  
    float x0,y0,h,x,z1,y1;  
    printf("Enter the value of x0,y0, h,x");  
    scanf( "%f %f %f %f", &x0 ,&y0 ,&h ,&x);  
    x1=x0; y1=y0;  
    while (1)  
    {  
        if (x1>x) return;  
        y1 +=h*df(x1,y1);  
        x1 +=h;  
        printf("When x = % 3 f " "y= % 4.2 f \n", x1,y1);  
    }  
}
```

Experiment No: 7

AIM : Flow chart & C program of Runge-Kutta Method

A) Flow chart



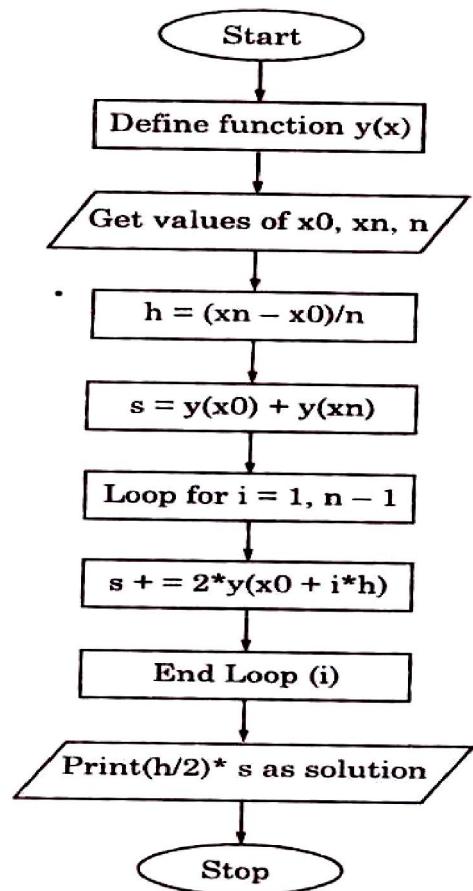
B) C- Program

```
/* Runge Kutta Method */
#include <stdio.h>
float f (float x, float y)
{
    return x+x*y;
}
main ( )
{
    float x0,y0,h,xn,x,y,k1,k2,k3,k4,k;
    printf("Enter the value of x0,y0, h, xn\n");
    scanf( "%f %f %f %f", &x0 ,&y0 ,&h ,&xn);
    x=x0; y=y0;
    while (1)
    {
        if (x == x ) break;
        k1= h*f( x, y );
        k2= h*f( x+h/2, y+k1/2 );
        k3= h*f( x+h/2, y+k2/2 );
        k4= h*f( x+h, y+k3 );
        k=(k1+(k2+k3)*2+k4)/6;
        x += h ; y += k ;
        printf("When x = % .4f " "y= % .4f \n", x,y);
    }
}
```

Experiment No: 8

AIM : Flow chart & C program of Trapezoidal Rule

A) Flow chart



B) C- Program

$y(x)$ is the function to be integrated

x_0 is x_0

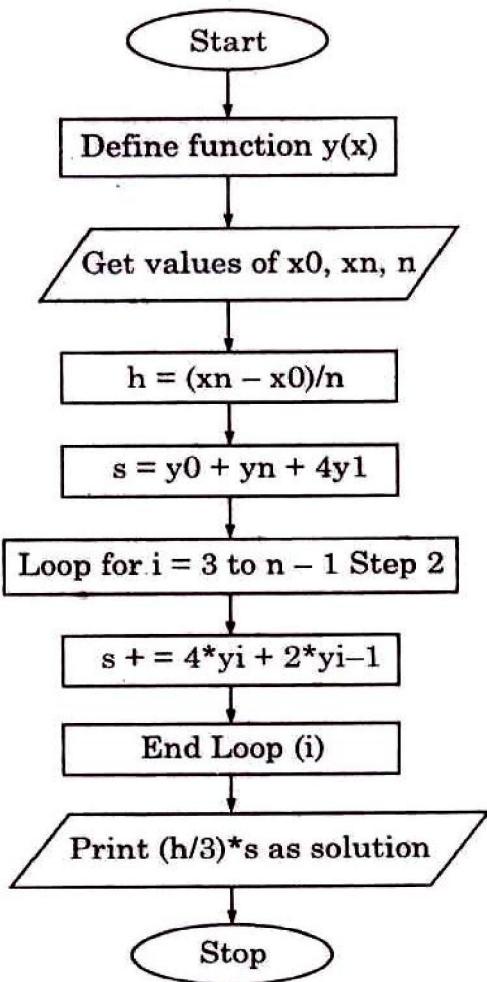
x_N is x_n

```
/* Trapezoidal Rule */
#include <stdio.h>
float y(float x)
{
    return 1/(1+x*x);
}
main()
{
    float x0,xn,h,s;
    int i, n;
    printf("Enter x0,xn, no. of subintervals");
    scanf ("%f %f %d", &x0 ,& xn ,& n );
    h=( xn- x0)/ n;
    s= y(x0) + y(xn) ;
    for (i=1;i<=n-1;i++)
        s += 2*y(x0+i*h);
    printf ("Value of integral is % .4 F \n" (h/2)*s) ;
}
```

Experiment No:9

AIM : Flow chart & C program of Simpson's 1/3rd Rule

A) Flow chart



B) C- Program

```
/* Simpson's 1/3rd Rule */
#include <stdio.h>
float y(float x)
{
    return 1/1(1+x*x);
}
main ( )
{
    float x0,xn,h,s;
    int I,n;
    printf("Enter x0,xn, no. of subintervals");
    scanf ( "%f %f %d ", &x0 ,& xn ,& n );
    h= ( xn- x0)/ n;
    s= y(x0) + y(xn) +4*y(x0+h);
    for (i=3;i<=n-1;i+=2)
        s += 4*y(x0+i*h)+2*y(x0+(i-1)*h);
    printf ( "Value of integral is % .4 F \n" (h/3)*s ) ;
}
```

Output - Enter x0, xn, no. of subintervals

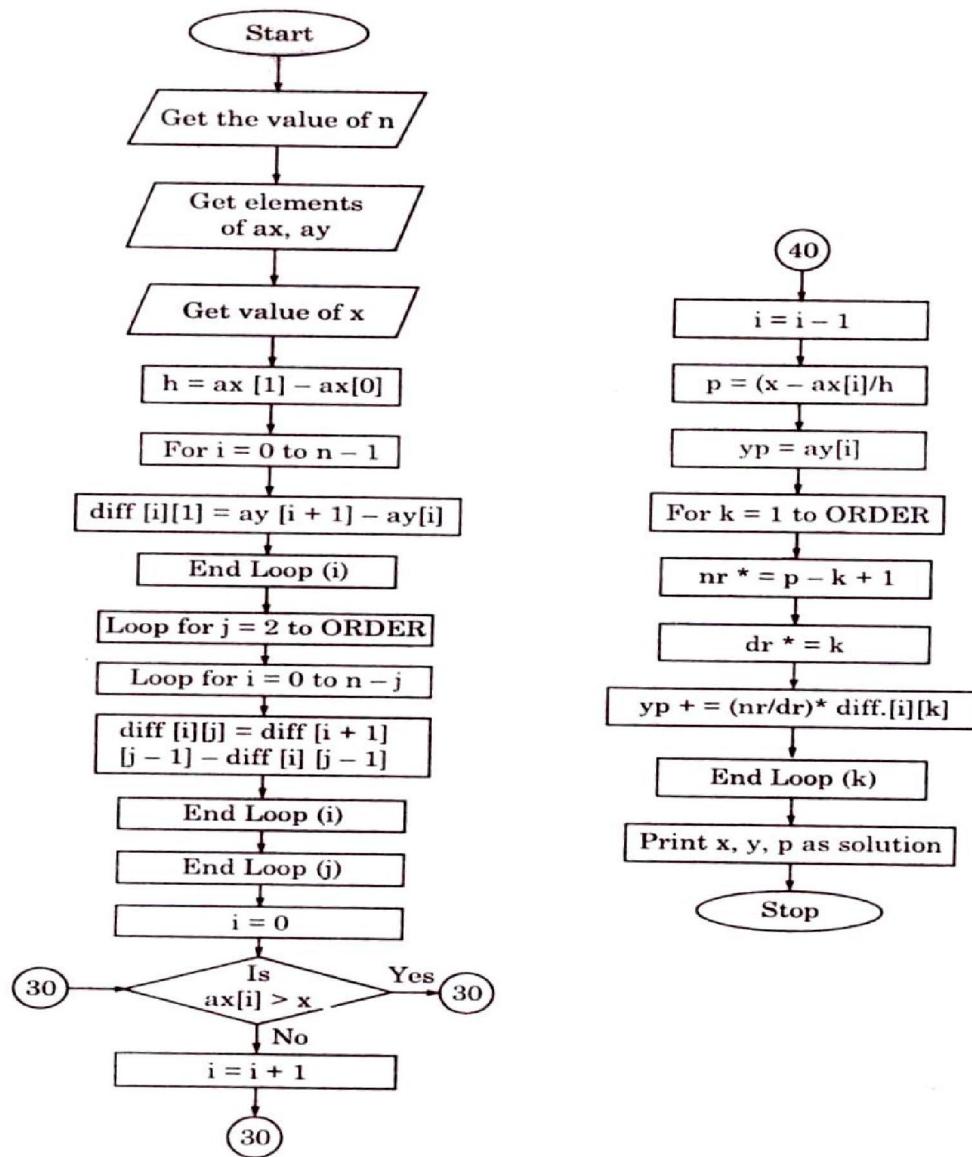
0 6 6

Value of integral is 1.3662

Experiment No: 10

AIM : Flow chart & C program of Newton Forward Method

A) Flow chart



B) C- Program

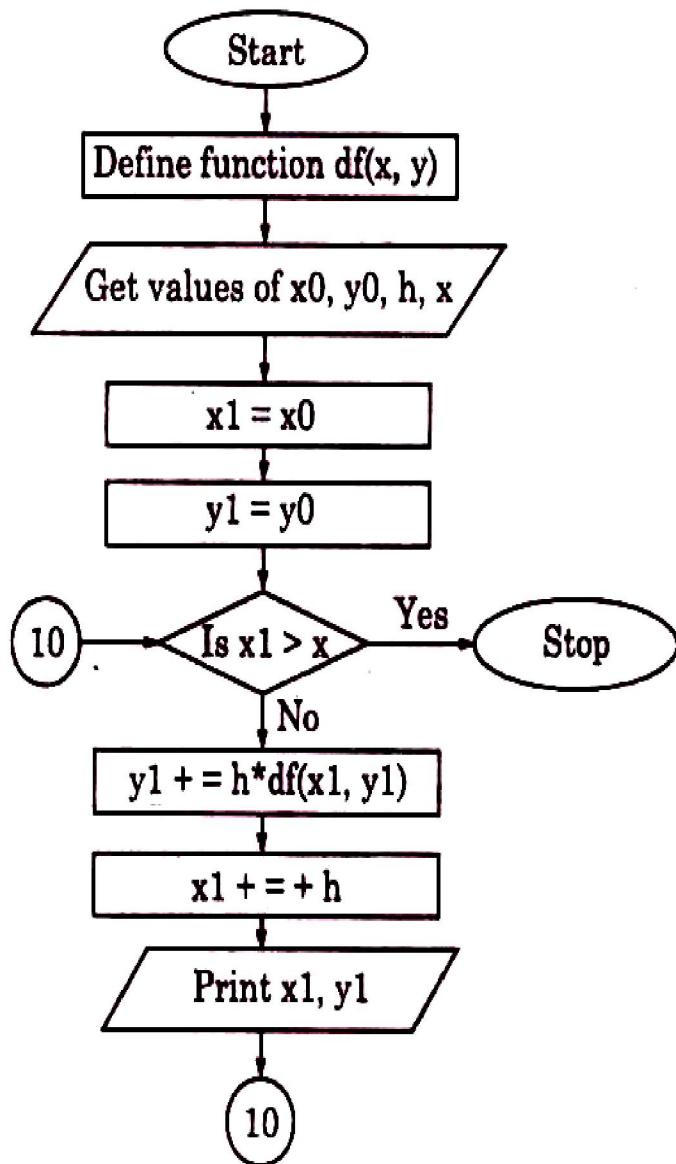
```
# include <studio .h>
#define MAXN 100
#define ORDER 4
main 0
{
float ax [MAXN+1] ,ay[MAXN+1],
diff [MAXN+1][ORDER +1],
nr-1.0,dr-1.0,x,p,h,yp;
int n,I,j,k;
print ("Enter the value of n/n");
scanf ("% d",&n);
print ("Enter the value in the form x,y\n");
for (i=0, i<-11,i++)
scanf ("% f % f",&ax[i], &ay[i]);
print ("Enter the values of x"
      for which value of y is wanted /n");
scanf ("% f",&x);
h=ax[1]-ax[0];
/*now making the diff. table*/
/* calculating the 1st order of difference */
for (i=0, i<=n-1;i++)
diff [i] [j] = ay [i/1]-ay[i];
/* calculating higher order differences */
for (j=2, j<=ORDER ; j++)
for (i=0,i<=n-j ; i++)
diff [i] [j] = diff [i+1] [j-1]-diff [i][j-1];
/* now finding x0 */
i= 0;
while (!(ax[i]>x))i++;
/* now ax[i] is x0 & xay [i] is y0 */
i--;
p-(x-ax[i])/h;yp-ay[i];
/* now carrying out interpolation */
for k=1; k<=ORDER; K++)
{
nr *-p k+1; dr * -k;
yp +-(m/n )* diff [i] [k],
}
print f ("when x= %6. If , y= %6.2f/n",x,yp);
}
```

Experiment No: 11

AIM

: Flow chart & C program of Modified Euler's Method

A) Flow chart



A) C- Program

```
/* Modified Euler's Method */
#include <stdio.h>
#include <conio.h>
#include <math.h>
void main ( )
{
float x[5], y[5], h, f(float, float);
int i, n;
clrscr ();
printf ("Modified Euler's Method\n\n");
printf("Enter the initial value of X, x(0)=\t");
scanf ("%f ", &x[0]);
printf("Enter the initial value of Y, y(0)=\t");
scanf ("%f ", &y[0]);
printf("Enter the width of interval, h=\t");
scanf ("%f ", &h);
printf("Enter the no. of steps, n=\t");
scanf ("%f ", &n);
printf("\n\n");
for(i=0;i<=n; i++)
{
y[i] = y[i-1] + h * f( x [ i-1 ] +h/2 , y[i-1] + ( h/2 ) * f( x[i-1], y[i-1]));
x[i] = x [ i-1 ] + h;
printf("x(%d)=%f\t y (%d)=%f\n" , i , x[i] ,i, y[i]);
}
float f(float x, float y );
{
float r, z ;
z= x*y ;
r= 2+ sqrt(z) ;
return(r);
}
```

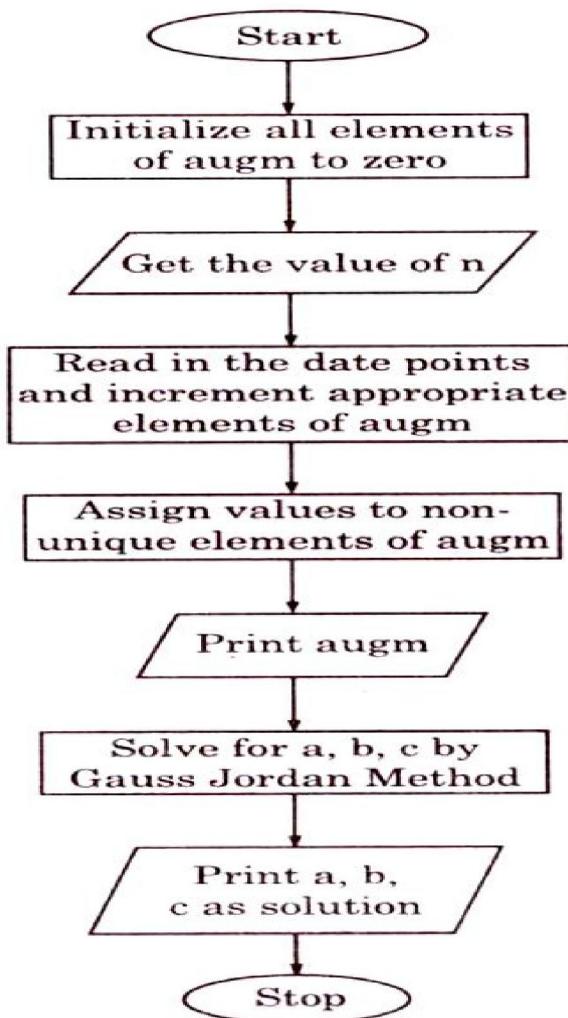
Out put

```
Enter the initial value of X, x (0) =1
Enter the initial value of Y, y (0) =1
Enter the width of interval, h= 2
Enter the no. of steps, n = 2
x (1)=1.20000      y(1)=1.639195
x (1)=1.40000      y(1)=2.359992
x (1)=1.60000      y(1)=3.165585
```

Experiment No: 12

AIM : Flow chart & C program of Least Square Method

A) Flow chart



B) C- Program

```
/* Parabolic fit by Least Square Method */
#include <stdio.h>
main ()
{
float augm [3] [4]={ { 0,0,0,0}, {0,0,0,0}, {0,0,0,0} };
float t, a, b, c, x, y, xsq;
int i, j, k , n;
printf ("Enter the no. of pairs of " "observed values." );
scanf ( "%d ",&n );
augm [0] [0] = n ;
for ( i = 0 ; i < n ; i++)
{
printf (" Pair no. %d \n",i+1);
scanf ( "%f %f" ,&x, &y );
xsq = x*x ;
augm [0] [1] += x ;
augm [0] [2] += xsq ;
augm [1] [2] += x * xsq;
augm [2] [2] += xsq * xsq;
augm [0] [3] += y ;
augm [1] [3] += x*y ;
augm [2] [3] += xsq*y
}
augm [1] [1] = augm [0] [2] ;
augm [2] [1] = augm [1] [2] ;
augm [1] [0] = augm augm [0] [1] ;
augm [2] [0] = augm augm [1] [1] ;
printf ("The augmented matrix is : -");
for ( i=0 ;i<3; i++)
{
for ( j=0 ;j<4; j++)
printf ("% 9.4 f" , augm [i] [j] );
printf ("\n" );
}
/* Now solving for a, b, c by Gauss jorden Method */
for ( j=0;j<3;j++)
for ( i=0; i<3; i++)
```

```
if (i!=j)
{
t= augm [i] [j] / augm [j] [j]
for (ki=0;k<4; k++)
augm [i] [k]-= augm [j] [k]*t;
}
a=
b=
c=
printf("a= % 8.4 f b = 8.4 f" "c =% 8.4 f\n", a ,b ,c );
}
```