NAME:-SAPTARSHI PANI

ROLL NUMBER:- 002010801148

DEPARTMENT:- ELECTRICAL ENGINEERING

SECTION:- B

GROUP:- 1

YEAR:- 1ST

SEMESTER:- 2ND

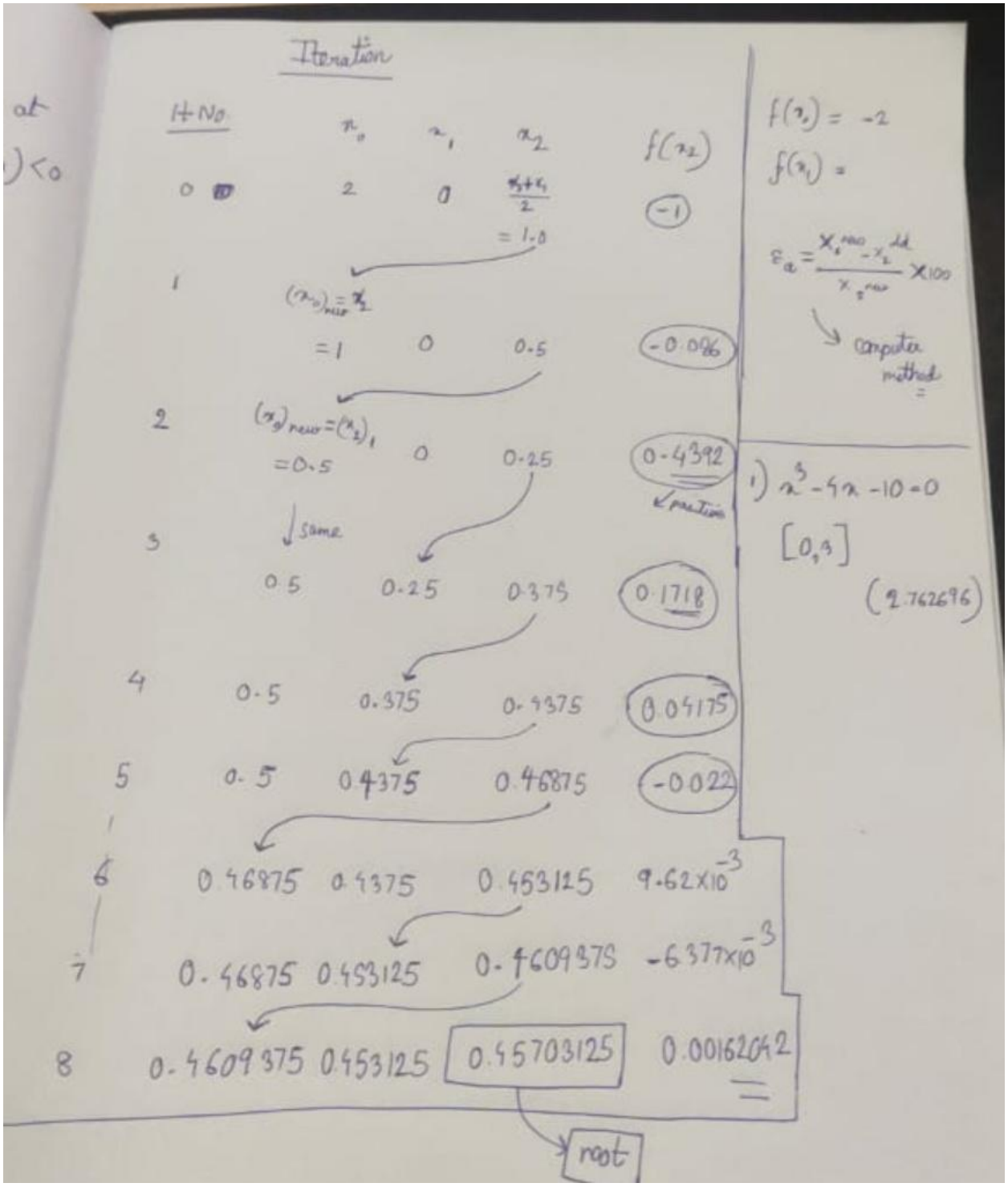
COMPUTER PROGRAMMING AND NUMERICAL LANGUAGES

2021-2022

PROBLEM 1:-

Problem:-
Solve,f(x)=2^x-3*x in the interval [0,2] and assume ε=0.01

Manual Solution:-



Source Code:-

```
# include <stdio.h>
# include <math.h>
int main()
{
    float x1,x2,e,z,fz,fx1,fx2,c1,c2,z1;
```

```
    int c;
    c=0;
    printf("Enter the lower limit \n");
    scanf("%f",&x1);
    printf("Enter the upper limit \n");
    scanf("%f",&x2);
    printf("Enter the user defined tolerance \n");
    scanf("%f",&e);
    fx1=pow(2,x1)-3*x1;
    fx2=pow(2,x2)-3*x2;
    if(fx1<fx2)
    {
        c1=x1;
        c2=x2;
    }
    else
    {
        c1=x2;
        c2=x1;
    }
    z=(c1+c2)/2;
    fz=pow(2,z)-3*z;
    if((fz*fx1)<0)
    c1=z;
    else
    c2=z;
    while(fabs(c1-c2)>e)
    {
        c=c+1;
        z=(c1+c2)/2;
        fz=pow(2,z)-3*z;
        if((fz*fx1)<0)
        c1=z;
        else
        c2=z;
    }
    z1=(c1+c2)/2;
    printf("The root of the equation as per bisection method is=%f
\n",z1);
    printf("The root resulted after %dth iteration",(c+1));
```

```
}
```

Output Terminal:-
Enter the lower limit
0
Enter the upper limit
2
Enter the user defined tolerance
0.01
The root of the equation as per bisection method is=0.457031
The root resulted after 8th iteration

PROBLEM 2:-

Problem:-

$$f(x) = x^2 - 25$$

Manual Solution:-

	Iteration No.	x_0	x_1	x_2	f_0	f_1	f_2
Initialisation		2.0	7.0		-21	24	
$x_0 \leftarrow x_2$	1	4.8	7.0	4.8	-1.96	24	-1.96
$x_1 \leftarrow x_2$	2	4.8	6.587	6.587	-1.96	18.38	18.38
$x_0 \leftarrow x_2$	3	4.9721	6.587	4.9721	-0.2782	18.38	-0.2782
$x_0 \leftarrow x_2$	4	4.9961	6.587	4.9961	-0.039	18.38	-0.039
$x_0 \leftarrow x_2$	5	4.994	6.587	4.994	-0.0053	18.38	-0.0053
$x_0 \leftarrow x_2$	6	4.9998	6.587	4.9998	-0.0019	18.38	-0.0014

Source Code:-

```
# include <stdio.h>

int main()
{
    int i,n,c;
```

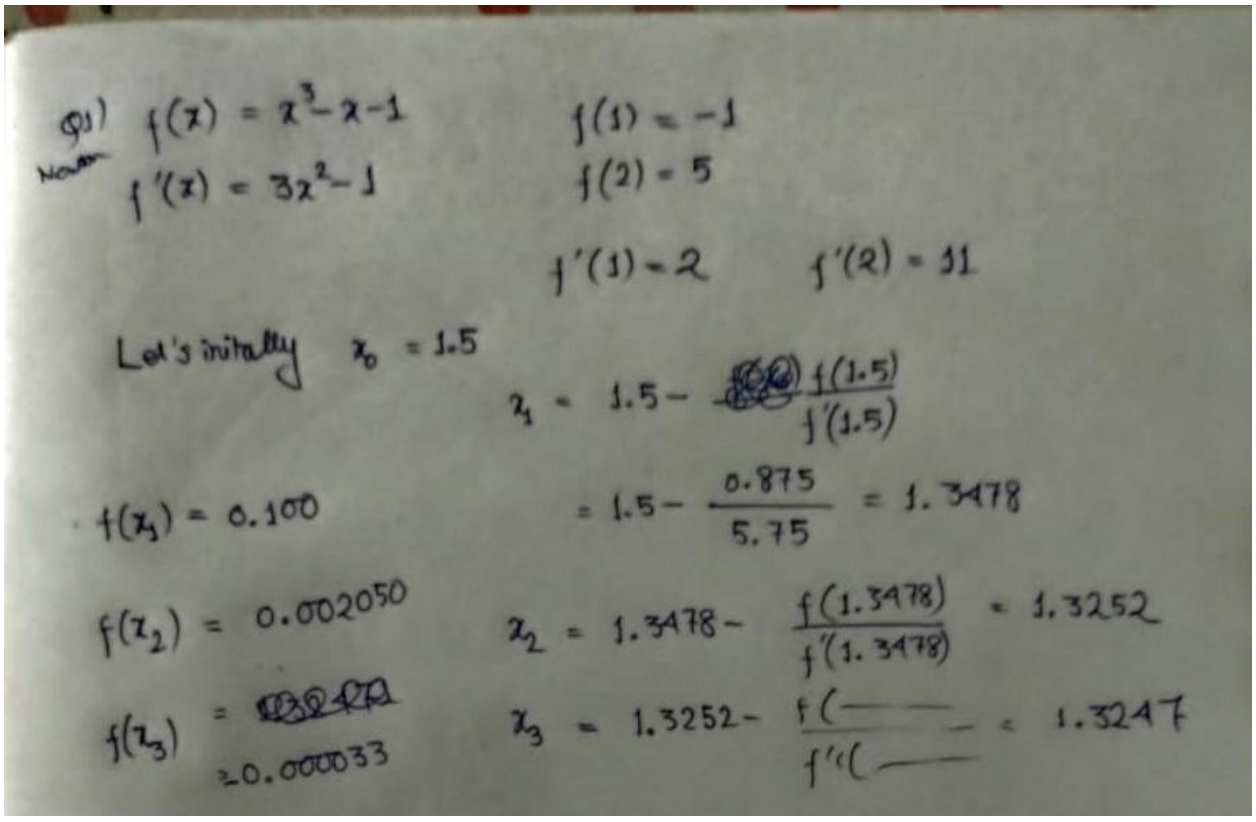
```
    c=0;
    float x1,x2,fx1,fx2,xm,fxm,c1,c2,fc1,fc2;
    printf("Enter the lower limit \n");
    scanf("%f",&x1);
    printf("Enter the upper limit \n");
    scanf("%f",&x2);
    printf("Enter the number of iterations \n");
    scanf("%d",&n);
    fx1=x1*x1-25;
    fx2=x2*x2-25;
    if(fx1<fx2)
        {
            c1=x1;
            c2=x2;
        }
    else
        {
            c2=x1;
            c1=x2;
        }
    fc1=c1*c1-25;
    fc2=c2*c2-25;
    xm=((c2*fc1)-(c1*fc2))/(fc1-fc2);
    fxm=xm*xm-25;
    if((fc1*fxm)<0)
        c2=xm;
    else
        c1=xm;
    for(i=1;i<=n;i=i+1)
        {
            c=c+1;
            fc1=c1*c1-25;
            fc2=c2*c2-25;
            xm=((c2*fc1)-(c1*fc2))/(fc1-fc2);
            fxm=xm*xm-25;
            if((fc1*fxm)<0)
            c2=xm;
            else
            c1=xm;
        }
```

```
    printf("The root of the equation as per Regula Falsi Method
is=%f\n",xm);
    printf("The root resulted after %dth iteration",c);
    return 0;
}
```

Output Terminal:-
Enter the lower limit
2
Enter the upper limit
7
Enter the number of iterations
6
The root of the equation as per Regula Falsi Method is=4.999985
The root resulted after 6th iteration

PROBLEM 3:-

Problem and Manual Solution:-



Source Code:-

```c
# include <stdio.h>

int main()
{
    float x1,x2;
    int n;
    printf("Enter the lower limit \n");
    scanf("%f",&x1);
    printf("Enter the upper limit \n");
    scanf("%f",&x2);
    printf("Enter the number of iterations \n");
    scanf("%d",&n);
    float x0;
    int i,c=0;
    x0=(x1+x2)/2;
    for(i=1;i<=n;i++)
    {
        c++;
        x0=x0-((x0*x0*x0-x0-1)/(3*x0*x0-1));
    }
    printf("The root of the equation as per Newton-Raphson is=%f\n",x0);
    printf("The root resulted after %dth iteration",c);
    return 0;
}
```

Output Terminal:-
Enter the lower limit
1
Enter the upper limit
2
Enter the number of iterations
3
The root of the equation as per Newton-Raphson is=1.324718
The root resulted after 3th iteration

PROBLEM 4:-

Problem:-

Example 1

The upward velocity of a rocket is given at three different times

Table 1 Velocity vs. time data.

Time, t (s)	**Velocity,** v (m/s)
5	106.8
8	177.2
12	279.2

The velocity data is approximated by a polynomial as:

$$v(t) = a_1 t^2 + a_2 t + a_3, \qquad 5 \le t \le 12.$$

Find the velocity at t=6 seconds .

Manual Solution:-

Example 1 Cont.

Assume

$$v(t) = a_1 t^2 + a_2 t + a_3, \quad 5 \le t \le 12.$$

Results in a matrix template of the form:

$$\begin{bmatrix} t_1^2 & t_1 & 1 \\ t_2^2 & t_2 & 1 \\ t_3^2 & t_3 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

Using data from Table 1, the matrix becomes:

$$\begin{bmatrix} 25 & 5 & 1 \\ 64 & 8 & 1 \\ 144 & 12 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 106.8 \\ 177.2 \\ 279.2 \end{bmatrix}$$

Forward Elimination: Step 1

$$\left[\begin{array}{ccc|c} 25 & 5 & 1 & 106.8 \\ 64 & 8 & 1 & 177.2 \\ 144 & 12 & 1 & 279.2 \end{array}\right]$$

Divide Equation 1 by 25 and multiply it by 64, $\frac{64}{25} = 2.56$

$$[25 \quad 5 \quad 1 \quad \vdots \quad 106.8] \times 2.56 = [64 \quad 12.8 \quad 2.56 \quad \vdots \quad 273.408]$$

Subtract the result from Equation 2

$$\begin{array}{r} [64 \quad 8 \quad 1 \quad \vdots \quad 177.2] \\ -[64 \quad 12.8 \quad 2.56 \quad \vdots \quad 273.408] \\ \hline [0 \quad -4.8 \quad -1.56 \quad \vdots \quad -96.208] \end{array}$$

Substitute new equation for Equation 2

$$\left[\begin{array}{ccc|c} 25 & 5 & 1 & 106.8 \\ 0 & -4.8 & -1.56 & -96.208 \\ 144 & 12 & 1 & 279.2 \end{array}\right]$$

Forward Elimination: Step 1 (cont.)

$$\left[\begin{array}{ccc|c} 25 & 5 & 1 & 106.8 \\ 0 & -4.8 & -1.56 & -96.208 \\ 144 & 12 & 1 & 279.2 \end{array}\right]$$

Divide Equation 1 by 25 and multiply it by 144, $\frac{144}{25} = 5.76$

$$[25 \quad 5 \quad 1 \quad \vdots \quad 106.8] \times 5.76 = [144 \quad 28.8 \quad 5.76 \quad \vdots \quad 615.168]$$

Subtract the result from Equation 3

$$\begin{array}{r} [144 \quad 12 \quad 1 \quad \vdots \quad 279.2] \\ -[144 \quad 28.8 \quad 5.76 \quad \vdots \quad 615.168] \\ \hline [0 \quad -16.8 \quad -4.76 \quad \vdots \quad -335.968] \end{array}$$

Substitute new equation for Equation 3

$$\left[\begin{array}{ccc|c} 25 & 5 & 1 & 106.8 \\ 0 & -4.8 & -1.56 & -96.208 \\ 0 & -16.8 & -4.76 & -335.968 \end{array}\right]$$

Forward Elimination: Step 2

$$\left[\begin{array}{ccc|c} 25 & 5 & 1 & 106.8 \\ 0 & -4.8 & -1.56 & -96.208 \\ 0 & -16.8 & -4.76 & -335.968 \end{array}\right]$$

Divide Equation 2 by −4.8 and multiply it by −16.8, $\frac{-16.8}{-4.8} = 3.5$

$$[0 \quad -4.8 \quad -1.56 \quad \vdots \quad -96.208] \times 3.5 = [0 \quad -16.8 \quad -5.46 \quad \vdots \quad -336.728]$$

Subtract the result from Equation 3

$$\begin{array}{r} [0 \quad -16.8 \quad -4.76 \quad \vdots \quad 335.968] \\ -[0 \quad -16.8 \quad -5.46 \quad \vdots \quad -336.728] \\ \hline [0 \quad 0 \quad 0.7 \quad \vdots \quad 0.76] \end{array}$$

Substitute new equation for Equation 3

$$\left[\begin{array}{ccc|c} 25 & 5 & 1 & 106.8 \\ 0 & -4.8 & -1.56 & -96.208 \\ 0 & 0 & 0.7 & 0.76 \end{array}\right]$$

Back Substitution

$$\left[\begin{array}{ccc|c} 25 & 5 & 1 & 106.8 \\ 0 & -4.8 & -1.56 & -96.2 \\ 0 & 0 & 0.7 & 0.7 \end{array}\right] \Rightarrow \begin{bmatrix} 25 & 5 & 1 \\ 0 & -4.8 & -1.56 \\ 0 & 0 & 0.7 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 106.8 \\ -96.208 \\ 0.76 \end{bmatrix}$$

Solving for a_3

$$0.7a_3 = 0.76$$

$$a_3 = \frac{0.76}{0.7}$$

$$a_3 = 1.08571$$

Back Substitution (cont.)

$$\begin{bmatrix} 25 & 5 & 1 \\ 0 & -4.8 & -1.56 \\ 0 & 0 & 0.7 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 106.8 \\ -96.208 \\ 0.76 \end{bmatrix}$$

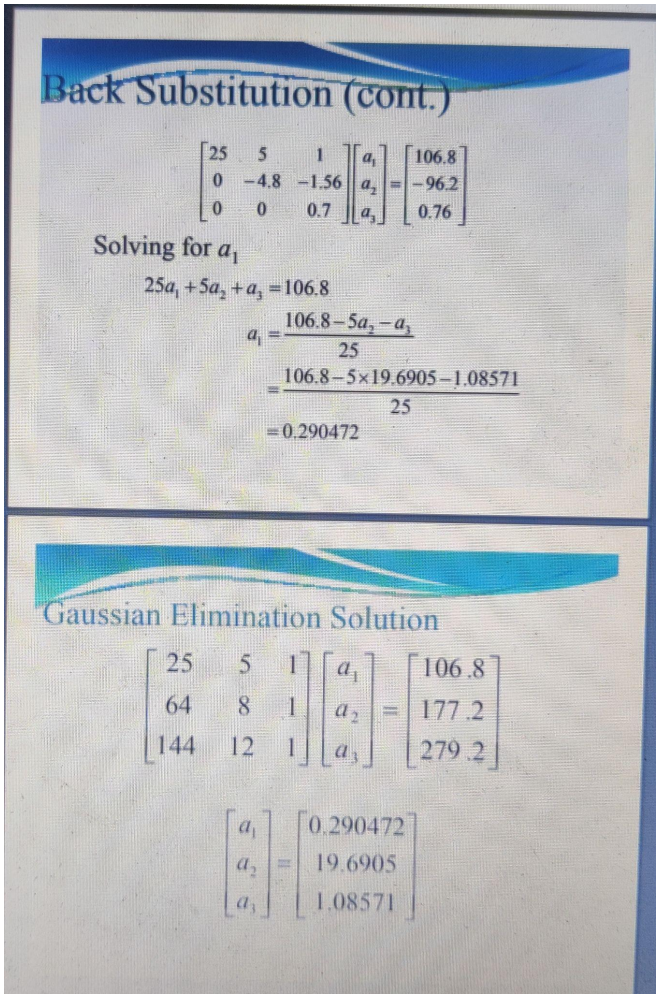
Solving for a_2

$$-4.8a_2 - 1.56a_3 = -96.208$$

$$a_2 = \frac{-96.208 + 1.56a_3}{-4.8}$$

$$a_2 = \frac{-96.208 + 1.56 \times 1.08571}{-4.8}$$

$$a_2 = 19.6905$$



Source Code:-

```
#include<stdio.h>
int main()
{
    int i,j,k,n;
    float arr[100][100],c,a[100],sum=0.0;
    printf("\nEnter the order of matrix: ");
    scanf("%d",&n);
    printf("\nEnter the elements of augmented matrix row-wise:\n\n");
    for(i=1; i<=n; i++)
    {
        for(j=1; j<=(n+1); j++)
        {
            printf("A[%d][%d] : ", i,j);
            scanf("%f",&arr[i][j]);
        }
```

```
    }
    /* forward elimination*/
    for(j=1; j<=n; j++)
    {
        for(i=1; i<=n; i++)
        {
            if(i>j)
            {
                c=arr[i][j]/arr[j][j];
                for(k=1; k<=n+1; k++)
                {
                    arr[i][k]=arr[i][k]-c*arr[j][k];
                }
            }
        }
    }
     a[n]=arr[n][n+1]/arr[n][n];
    /* back substitution*/
    for(i=n-1; i>=1; i--)
    {
        sum=0;
        for(j=i+1; j<=n; j++)
        {
            sum=sum+arr[i][j]*a[j];
        }
        a[i]=(arr[i][n+1]-sum)/arr[i][i];
    }
    printf("\nThe solution is: \n");
    for(i=1; i<=n; i++)
    {
        printf("\nx%d=%f\t",i,a[i]); /* solution matrix*/
    }
    return 0;
}
```

Output Terminal:-
Enter the order of matrix: 3

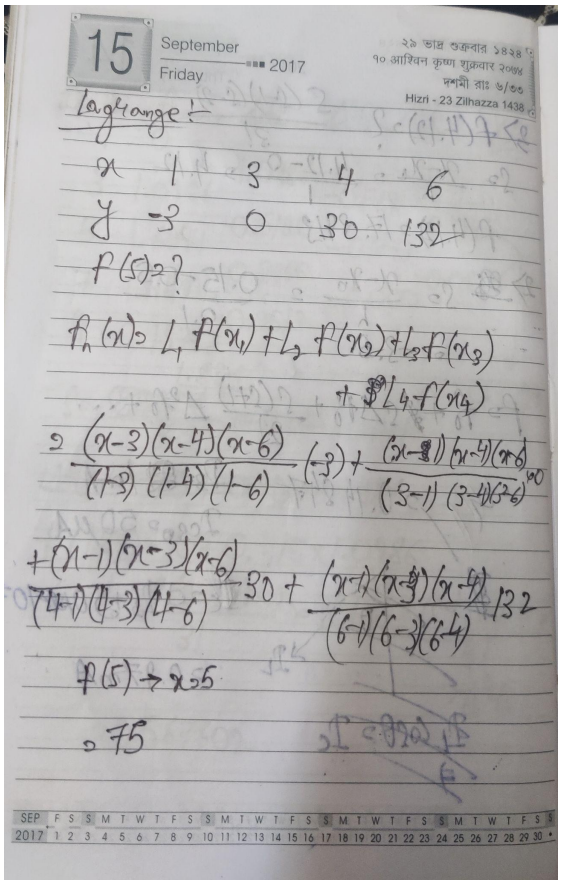
Enter the elements of augmented matrix row-wise:

A[1][1] : 25
A[1][2] : 5
A[1][3] : 1
A[1][4] : 106.8
A[2][1] : 64
A[2][2] : 8
A[2][3] : 1
A[2][4] : 177.2
A[3][1] : 144
A[3][2] : 12
A[3][3] : 1
A[3][4] : 279.2
The solution is:

x1=0.290478
x2=19.690453
x3=1.085793

PROBLEM 5:-

Problem and Manual Solution:-



Source Code:-

```c
# include<stdio.h>

int main()
{
    int n;
    float a;
    printf("Enter the number of user-input data\n");
    scanf("%d",&n);
    float x[n];
    float y[n];
    printf("Enter the value of x for which f(x) is to be evaluated\n");
    scanf("%f",&a);
    int i,i1,j;
    float k,s;
    s=0;
    for(i=0;i<n;i++)
    {
        printf("Enter x%d\n",(i+1));
        scanf("%f",&x[i]);
        printf("Enter fx(%d)\n",(i+1));
        scanf("%f",&y[i]);
    }
    for(j=0;j<n;j++)
    {
        k=y[j];
        for(i1=0;i1<n;i1++)
        {
            if(i1!=j)
            {
                k=k*((a-x[i1])/(x[j]-x[i1]));
            }
        }
        s=s+k;
    }
    printf("f(%f)=%f\n",a,s);
    return 0;
}
```

Output Terminal:-

Enter the number of user-input data
4
Enter the value of x for which f(x) is to be evaluated
5
Enter x1
1
Enter fx(1)
-3
Enter x2
3
Enter fx(2)
0
Enter x3
4
Enter fx(3)
30
Enter x4
6
Enter fx(4)
132
f(5.000000)=75.000000

PROBLEM 6:-

Problem:-
Basically, I used a switch-case for user choice whether he/she wants the result according to Newton Forward or Backward Polynomial.

Source Code:-

```
#include<stdio.h>
#include<math.h>
int fact(int);
float make1(float,int);
float make2(float,int);
int main(){

printf("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~\n");
    printf("Newton'S Interpolation Method\n");

printf("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~\n\n\n");
    int n;
```

```
    printf("Enter the number of sample data to be entered: ");
    scanf("%d",&n);

    float x_val[n];
    float fx_val[n][n];

    int i,j=0;

    for(i=0;i<n;i++){
        printf("\nEnter x value: ");
        scanf("%f",&x_val[i]);
        printf("\nEnter corresponding f(x) value: ");
        scanf("%f",&fx_val[0][i]);

    }

    float x;
    printf("\nEnter the interpolating point:");
    scanf("%f",&x);
    int c=10;

    float s1,s2;
    while(c!=0){
        printf("\nEnter\n 1 for Forward Elimination Method \n 2 for
Backward Elimination \n 0 to exit.");
        scanf("%d",&c);
        switch(c){
            case 0:
                break;
            case 1:
                for(i=0;i<(n-1);i++){
                    for(j=0;j<(n-i-1);j++){
                        fx_val[i+1][j]=fx_val[i][j+1]-fx_val[i][j];
                    }
                }
                 printf("\n\n\n");
                for(i=0;i<n;i++){
                    printf("\n");
                    for(j=0;j<(n-i);j++){
                        printf("%f  ",fx_val[j][i]);
```

```
                    }
                }
                 s1=(x-x_val[0])/(x_val[1]-x_val[0]);


                float in1=fx_val[0][0];
                for(i=0;i<n-1;i++){
                    in1=in1+(make1(s1,(i+1))/fact(i+1))*fx_val[i+1][0];
                }
                printf("\n\nThe interpolated value is: %f\n\n",in1);
                break;

            case 2:
                for(i=0;i<(n-1);i++){
                    for(j=0;j<(n-i-1);j++){
                        fx_val[i+1][j]=fx_val[i][j+1]-fx_val[i][j];
                    }
                }

                for(i=0;i<n;i++){
                    printf("\n");
                    for(j=0;j<(n-i);j++){
                        printf("%f  ",fx_val[j][i]);
                    }
                }


                 s2=(x-x_val[n-1])/(x_val[1]-x_val[0]);
                float in2=fx_val[0][n-1];
                for(i=0;i<n-1;i++){

in2=in2+(make2(s2,(i+1))/fact(i+1))*fx_val[i+1][n-i-2];
                }
                printf("\n\nThe interpolated value is: %f\n\n",in2);
                break;

            }
        }
    }
```

```
int fact(int x){
    int f=1;
    int i=0;
    for(i=1;i<=x;i++){
        f=f*i;
    }
    return f;
}

float make1(float s, int n){
    int i=0;
    float r=1;

    for(i=0;i<n;i++){
        r=r*(s-i);
    }
    return r;
}

float make2(float s, int n){
    int i=0;
    float r=1;

    for(i=0;i<n;i++){
        r=r*(s+i);
    }
    return r;
}
```

Output Terminal:-
Enter the number of data points and the data points and then select Newton Forward or Backward Polynomial. You will get the desired result.

PROBLEM 7:-

Problem:-

5. Fit a straight line to the following data:(7)

x:	0	1	2	3	4
f(x):	1	1.8	3.3	4.5	6.3

Manual Solution:-

x	$f(x)$	x^2	$xf(x)$
0	1	0	0
1	1.8	1	1.8
2	3.3	4	6.6
3	4.5	9	13.5
4	6.3	16	25.2
10	16.9	30	47.1

We can fit this data in a straight line of the form

$y = a_0 + a_1 \cdot x$

where, $a_0 = \dfrac{\sum_{i=1}^{5} x_i^2 \sum_{i=1}^{5} f(x_i) - \sum_{i=1}^{5} x_i \sum_{i=1}^{5} x_i f(x_i)}{5\sum_{i=1}^{5} x_i^2 - \left(\sum_{i=1}^{5} x_i\right)^2}$

$\Rightarrow a_0 = \dfrac{16.9 \times 30 - 10 \times 47.1}{5 \times 30 - (10)^2} = 0.72$

$a_1 = \dfrac{5\sum_{i=1}^{5} x_i f(x_i) - \sum_{i=1}^{5} x_i \sum_{i=1}^{5} f(x_i)}{5\sum_{i=1}^{5} x_i^2 - \left(\sum_{i=1}^{5} x_i\right)^2}$

$= \dfrac{5 \times 47.1 - 10 \times 16.9}{5 \times 30 - (10)^2} = 1.33$

So, the straight line is $y = 0.72 + 1.33x$

Source Code:-

```c
# include <stdio.h>

int main()
{
    int n,i;
    float sx2=0,sxy=0;
    float sx=0,sy=0;
    float a0,a1;
    printf("Enter the number of points\n");
    scanf("%d",&n);
    float x[n];
    float y[n];
    printf("Enter the x co-ordinates one by one\n");
    for(i=0;i<n;i++)
    {
        scanf("%f",&x[i]);
    }
    printf("Enter the y coordinates one by one\n");
    for(i=0;i<n;i++)
    {
        scanf("%f",&y[i]);
    }
    for(i=0;i<n;i++)
    {
        sx=sx+x[i];
        sy=sy+y[i];
        sx2=sx2+(x[i]*x[i]);
        sxy=sxy+(x[i]*y[i]);
    }
    a0=((sx2*sy)-(sx*sxy))/((n*sx2)-(sx*sx));
    a1=((n*sxy)-(sx*sy))/((n*sx2)-(sx*sx));
    printf("The equation of straight line as per least square method
is:\n");
    printf("y=%f+%fx",a0,a1);
    return 0;
}
```

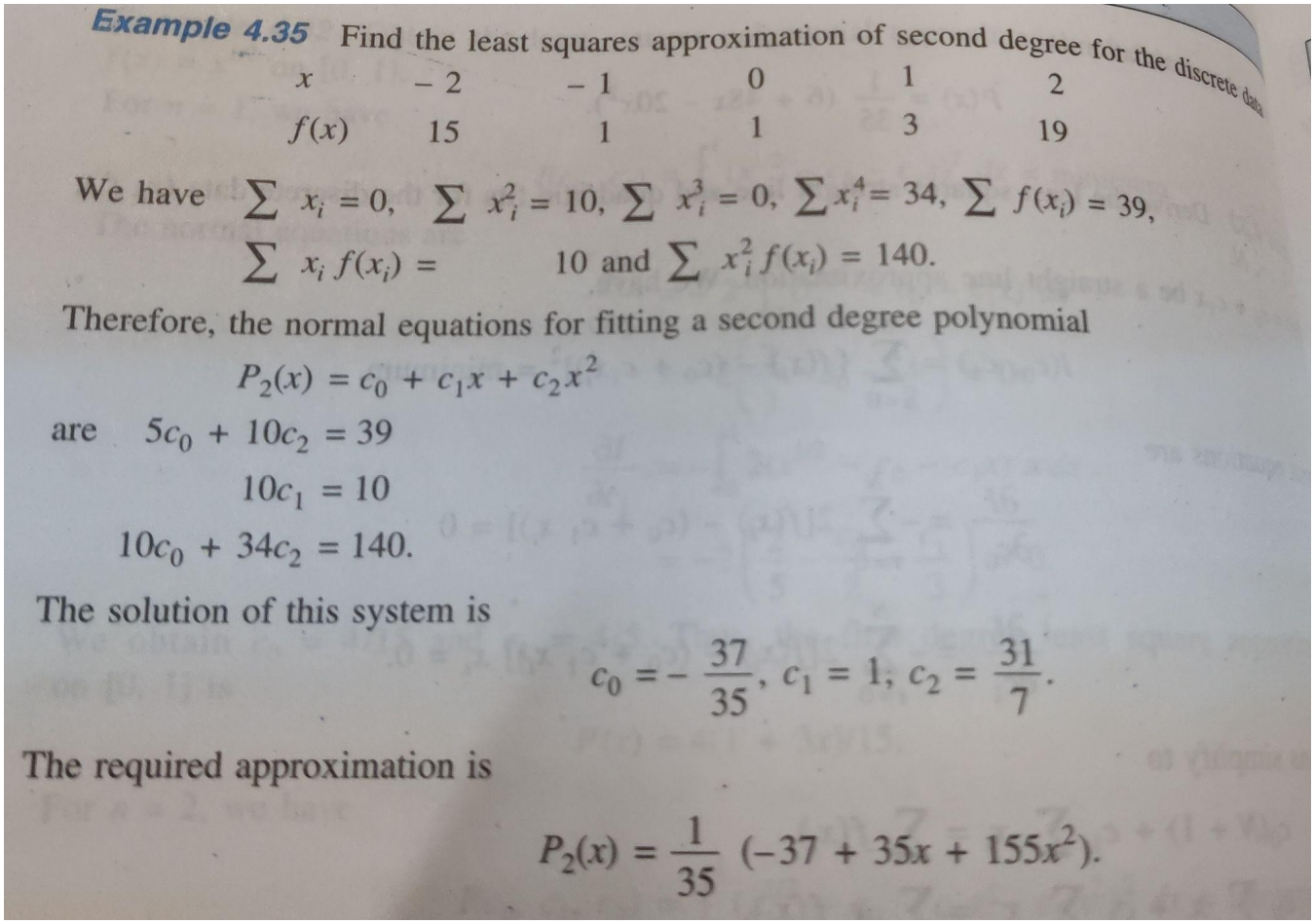
Output Terminal:-

Enter the number of points

5
Enter the x co-ordinates one by one
0
1
2
3
4
Enter the y coordinates one by one
1
1.8
3.3
4.5
6.3
The equation of straight line as per least square method is:
y=0.720001+1.330000x

PROBLEM 8:-

Problem and Manual Solution:-



Example 4.35 Find the least squares approximation of second degree for the discrete data

x	-2	-1	0	1	2
$f(x)$	15	1	1	3	19

We have $\sum x_i = 0$, $\sum x_i^2 = 10$, $\sum x_i^3 = 0$, $\sum x_i^4 = 34$, $\sum f(x_i) = 39$,
$\sum x_i f(x_i) =$ 10 and $\sum x_i^2 f(x_i) = 140$.

Therefore, the normal equations for fitting a second degree polynomial

$$P_2(x) = c_0 + c_1 x + c_2 x^2$$

are $5c_0 + 10c_2 = 39$

$$10c_1 = 10$$

$$10c_0 + 34c_2 = 140.$$

The solution of this system is

$$c_0 = -\frac{37}{35},\ c_1 = 1,\ c_2 = \frac{31}{7}.$$

The required approximation is

$$P_2(x) = \frac{1}{35}(-37 + 35x + 155x^2).$$

Source Code:-

```
# include<stdio.h>

int main()
{
    int n,i;
    float sx2=0,sxy=0;
    float sx=0,sy=0,sx3=0,sx4=0,sxy2=0;
    float a0,a1;
    printf("Enter the number of points\n");
    scanf("%d",&n);
    float x[n];
    float y[n];
    printf("Enter the x co-ordinates one by one\n");
    for(i=0;i<n;i++)
    {
        scanf("%f",&x[i]);
    }
    printf("Enter the y coordinates one by one\n");
    for(i=0;i<n;i++)
    {
        scanf("%f",&y[i]);
    }
    for(i=0;i<n;i++)
    {
        sx=sx+x[i];
        sx2=sx2+(x[i]*x[i]);
        sy=sy+y[i];
        sx3=sx3+(x[i]*x[i]*x[i]);
        sxy=sxy+(x[i]*y[i]);
        sx4=sx4+(x[i]*x[i]*x[i]*x[i]);
        sxy2=sxy2+(x[i]*x[i]*y[i]);
    }
    float arr[100][100];
    float a[100];
    float c,sum=0.0;
    int j,k;
    arr[1][1]=(float)n;
    arr[1][2]=sx;
    arr[1][3]=sx2;
```

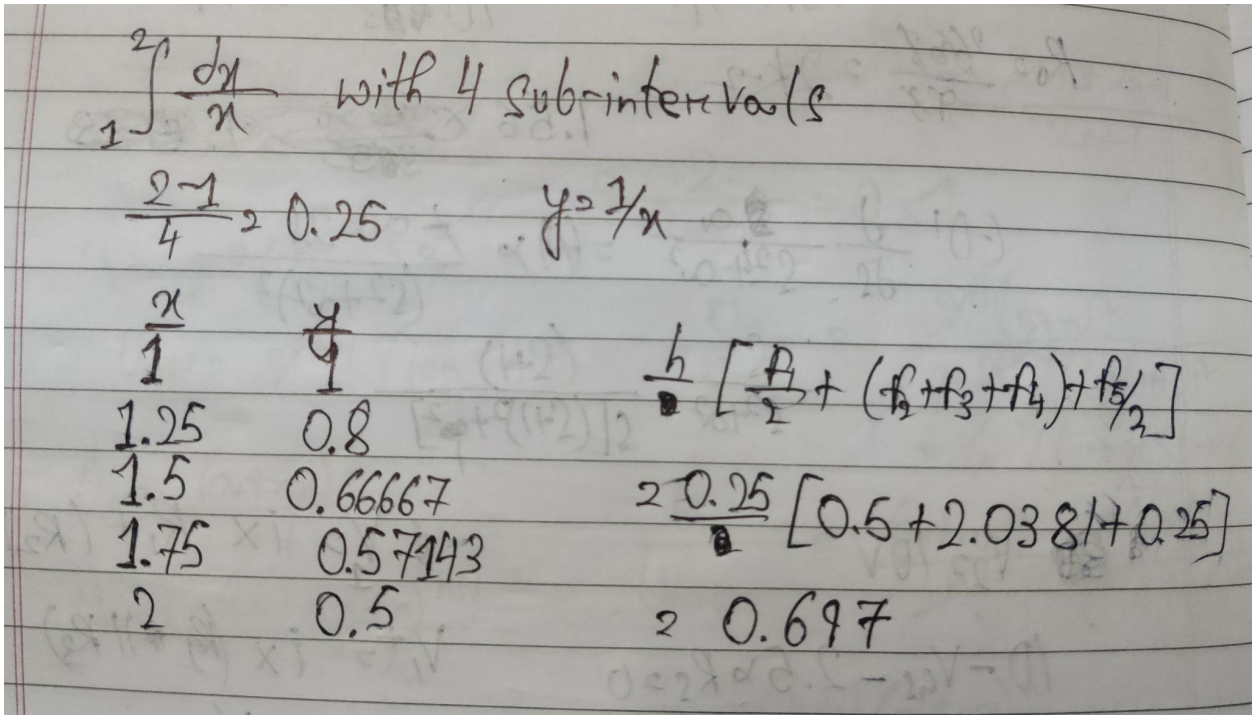
```
    arr[1][4]=sy;
    arr[2][1]=sx;
    arr[2][2]=sx2;
    arr[2][3]=sx3;
    arr[2][4]=sxy;
    arr[3][1]=sx2;
    arr[3][2]=sx3;
    arr[3][3]=sx4;
    arr[3][4]=sxy2;
    for(j=1; j<=3; j++)
    {
        for(i=1; i<=3; i++)
        {
            if(i>j)
            {
                c=arr[i][j]/arr[j][j];
                for(k=1; k<=4; k++)
                {
                    arr[i][k]=arr[i][k]-c*arr[j][k];
                }
            }
        }
    }
     a[3]=arr[3][4]/arr[3][3];
     for(i=2; i>=1; i--)
    {
        sum=0;
        for(j=i+1; j<=3; j++)
        {
            sum=sum+arr[i][j]*a[j];
        }
        a[i]=(arr[i][4]-sum)/arr[i][i];
    }
    printf("The equation of polynomial as per least square method is:\n");
    printf("y=%f+%fx+%fx^2",a[1],a[2],a[3]);
    return 0;
}
```

Output Terminal:-
Enter the number of points

5
Enter the x co-ordinates one by one
-2
-1
0
1
2
Enter the y coordinates one by one
15
1
1
3
19
The equation of polynomial as per least square method is:
y=-1.057143+1.000000x+4.428571x^2

PROBLEM 9:-

Problem and Manual Solution:-



Source Code:-

```c
# include<stdio.h>

int main()
{
```

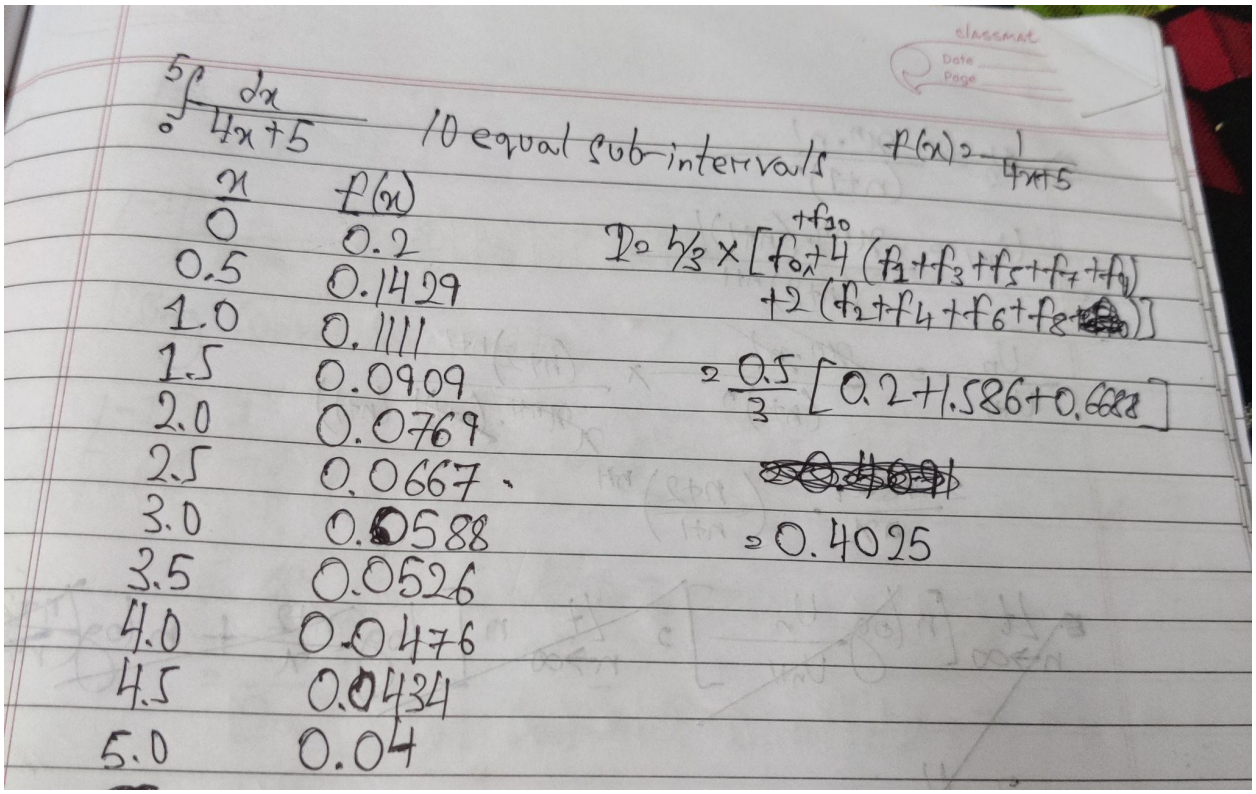
```
    float l,u;
    int n,i;
    printf("Enter the lower limit of integral\n");
    scanf("%f",&l);
    printf("Enter the upper limit of integral\n");
    scanf("%f",&u);
    printf("Enter the number of sub-intervals for the method\n");
    scanf("%d",&n);
    float s=0,s1;
    float x[(n+1)];
    float y[(n+1)];
    float h=(u-l)/n;
    for(i=0;i<=n;i++)
    {
        x[i]=l+i*h;
        y[i]=(1/x[i]);
    }
    for(i=0;i<=n;i++)
    {
        if((i==0)||(i==n))
        {
            s=s+(y[i]/2);
        }
        else
        {
            s=s+y[i];
        }
    }
    s1=h*s;
    printf("The integral of the function as per Trapezoidal Rule in given
range is=%f",s1);
    return 0;
}
```

Output Terminal:-
Enter the lower limit of integral
1
Enter the upper limit of integral
2
Enter the number of sub-intervals for the method
4

The integral of the function as per Trapezoidal Rule in given range is=0.697024

PROBLEM 10:-

Problem and Manual Solution:-



Source Code:-

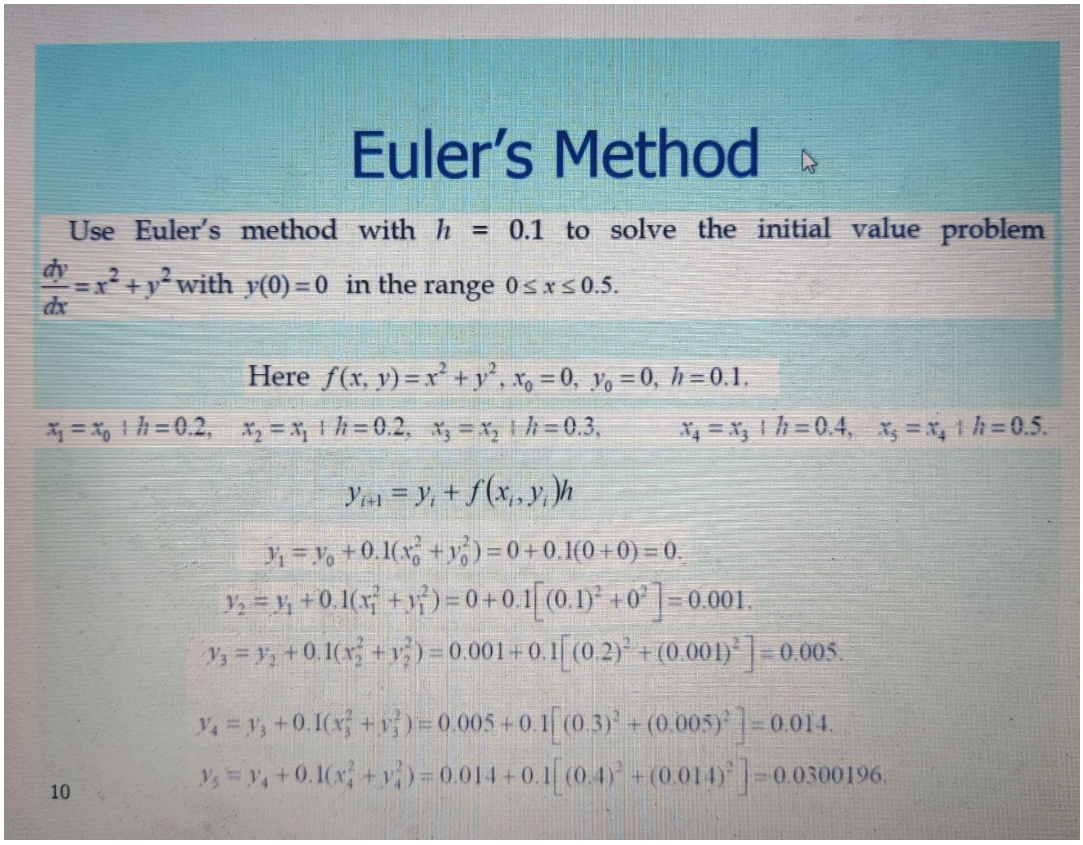
```
# include<stdio.h>

int main()
{
    float l,u;
    int n,i;
    printf("Enter the lower limit of integral\n");
    scanf("%f",&l);
    printf("Enter the upper limit of integral\n");
    scanf("%f",&u);
    printf("Enter the number of sub-intervals for the method\n");
    scanf("%d",&n);
    float s=0,s1;
    float x[(n+1)];
    float y[(n+1)];
```

```
    float h=(u-l)/(float)n;
    for(i=0;i<=n;i++)
    {
        x[i]=l+(i*h);
        y[i]=(1/(4*x[i]+5));
    }
    for(i=0;i<=n;i++)
    {
        if(i==0||i==n)
        {
            s=s+y[i];
        }
        else if((i%2)==0)
        {
            s=s+(2*y[i]);
        }
        else
        {
            s=s+(4*y[i]);
        }
    }
    s1=(h/3)*s;
    printf("The integral of the function as per Simpson's one-third Rule
in given range is=%f",s1);
    return 0;
}
```

Output Terminal:-
Enter the lower limit of integral
0
Enter the upper limit of integral
5
Enter the number of sub-intervals for the method
10
The integral of the function as per Simpson's one-third Rule in given range is=0.402521

PROBLEM 11:-

Problem and Manual Solution:-



Source Code:-

```
# include<stdio.h>

float fxy(float,float);
int main()
{
    float x1,x2;
    float y0;
    float h,w;
    int i;
    printf("Enter the lower limit of x\n");
    scanf("%f",&x1);
    printf("Enter the upper limit of x\n");
    scanf("%f",&x2);
    printf("Enter the initial value of y\n");
    scanf("%f",&y0);
    printf("Enter the value of h\n");
    scanf("%f",&h);
    w=(x2-x1)/h;
```

```
    float x=x1,y=y0;
    for(i=1;i<=w;i++)
    {
        y=y+h*fxy(x,y);
        x=x+h;
        printf("y%d=%f\n",i,y);
    }
    return 0;
}
float fxy(float a,float b)
{
    return ((a*a)+(b*b));
}
```

Output Terminal:-
Enter the lower limit of x
0
Enter the upper limit of x
0.5
Enter the initial value of y
0
Enter the value of h
0.1
y1=0.000000
y2=0.001000
y3=0.005000
y4=0.014003
y5=0.030022

PROBLEM 12:-

Problem:-
f(x)=(y*y-x*x)/(y*y+x*x)

Source Code:-

```
# include<stdio.h>

#define f(x,y) (y*y-x*x)/(y*y+x*x)

int main()
{
    float x0, y0, xn, h, yn, k1, k2, k3, k4, k;
```

```
    int i, n;
    printf("Enter the initial value of x\n");
    scanf("%f",&x0);
    printf("Enter the initial value of y\n");
    scanf("%f",&y0);
    printf("Enter the calculation point\n");
    scanf("%f",&xn);
    printf("Enter the number of iterations\n");
    scanf("%d",&n);
    h = (xn-x0)/n;
    printf("\n x0\ty0\tyn\n");
    printf("------------------\n");
    for(i=0; i < n; i++)
 {
  k1 = h * (f(x0, y0));
  k2 = h * (f((x0+h/2), (y0+k1/2)));
  k3 = h * (f((x0+h/2), (y0+k2/2)));
  k4 = h * (f((x0+h), (y0+k3)));
  k = (k1+2*k2+2*k3+k4)/6;
  yn = y0 + k;
  printf("%f\t%f\t%f\n",x0,y0,yn);
  x0 = x0+h;
  y0 = yn;
 }
 printf("\nValue of y at x =%f is %f",xn,yn);
    return 0;
}
```

Output Termnal:-

```
Enter the initial value of x
0
Enter the initial value of y
1
Enter the calculation point
0.6
Enter the number of iterations
3

 x0     y0      yn
------------------
0.000000        1.000000        1.196000
```

0.200000 1.196000 1.375267
0.400000 1.375267 1.533114

Value of y at x =0.600000 is 1.533114

CONCLUSION:-

MY SOURE CODES ARE DIRECTLY AVAILABLE ON THE FOLLOWING GIVEN LINK
https://github.com/SaptarshiPani/SEM-2