# **Assignment 1:**

Write a program in C to calculate f (0.5) by Newton's Forward Interpolation formula using the following data:

x	0	1	2	3
f(x)	1	0	1	10

**Answer: 0.625** 

Algorithm for Newton's Forward Interpolation formula:

- 1. Start
- 2. Read n
- 3. For i=0 to (n-1) do

Read xi

Next i

4. For i=0 to (n-1) do

Read yi,0

Next i

- 5. Read X
- 6. For j=1 to (n-1) do

For i=j to (n-1) do

yi,j =yi,j-1 - yi-1,j-1

Next i

Next j

7. For i=0 to (n-1) do

For j=0 to i do

Print yi,j

Next j

Next i

- 8. h = x1-x0
- 9. u = (X-x0)/h
- 10. term = 1.0
- 11. Sum = y0,0
- 12. For i=1 to (n-1) do

term = term \* (u-i+1)/i

Sum = sum + term \* yi,i

Next i

- 13. Print Sum
- 14. Stop

#### Program for Newton's Forward Interpolation formula

```
/*program for newton's forward interpolation*/
1.
2.
        #include<stdio.h>
3.
       #include<stdlib.h>
4.
       int main()
5.
       {
                float x[20],y[20][20],s,t=1.0,X,h,u;
6.
       int i,j,n;
7.
       printf("enter n:");
8.
       scanf("%d",&n);
9.
       for(i=0;i<n;i++)
10.
       {
       printf("enter x%d:",i+1);
11.
12.
       scanf("%f",&x[i]);
13.
       printf("enter y%d:",i+1);
14.
       scanf("%f",&y[i][0]);
15.
       }
16.
       printf("enter the point of interpolation:");
17.
       scanf("%f",&X);
18.
       for(j=1;j<n;j++)
19.
       {
20.
       for(i=j;i<n;i++)
21.
       y[i][j]=y[i][j-1]-y[i-1][j-1];
22.
23.
       printf("difference table\n\n");
24.
       for(i=0;i<n;i++)
25.
       {
26.
       for(j=0;j<=i;j++)
27.
       printf("%7.3f",y[i][j]);
28.
       printf("\n");
29.
       }
       h=x[1]-x[0];
30.
31.
       u=(X-x[0])/h;
       s=y[0][0];
32.
       for(i=1;i<n;i++)
33.
34.
35.
       t=t*(u-i+1)/i;
36.
       s=s+t*y[i][i];
37.
       }
38.
       printf("\nvalue of the function:");
39.
        printf("at x=%f is %0.3f",X,s);
40.
       return 0;}
```

```
Output:
enter n:4
enter x1:0
enter y1:1
enter x2:1
enter y2:0
enter x3:2
enter y3:1
enter x4:3
enter y4:10
enter the point of interpolation:0.5
difference table
 1.000
 0.000 - 1.000
 1.000 1.000 2.000
10.000 9.000 8.000 6.000
value of the function: at x=0.500000 is
0.625
Process exited after 64.99 seconds
with return value 0
Press any key to continue . . .
```

# Assignment 2:

Write a program in C to calculate f(2.5) by Newton's Backward Interpolation formula using the following data:

X	0	1	2	3
f(x)	1	0	1	10

Answer: 4.125

Algorithm for by Newton's Backward Interpolation formula:

- 1. Start
- 2. Read n
- 3. For i=0 to (n-1) do

Read xi

Next i

4. For i = 0 to (n-1) do

Read yi,0

Next i

5. h=x1-x0

6. For i=1 to (n-2) do

If  $x(i+1)-x(i)! \neq h$ 

Then flag =0

End of if

End of flag

7. If flag ==1

Print (" formula is applicable")

- 8. Read X
- 9. For j=1 to (n-1) do

For i=j to (n-1) do

yi,j= yi,j-1 - yi-1,j-1

Next i

Next j

10. For i=0 to (n-1) do

For j=0 to 1 do

Print yi,j

Next j

Next i

11. u = (X-x[n-1])/h

12. Term = 1.0

13. Sum= y n-1,0

14. For i=1 to (n-1) do

Term = term \* (u+i-1)/i

Sum = sum +t\* y n-1,i

Next i

15. Print Sum

16. Stop

```
Program for Newton's Backward Interpolation formula:
1. /*program for newton's backward interpolation*/
                                                                     Output
2. #include<stdio.h>
                                                                     enter n:4
3. #include<stdlib.h>
4. int main()
                                                                     enter x1:0
5. {
                                                                     enter y1:1
6. float x[20],y[20][20],s,t=1.0,X,h,u;
                                                                     enter x2:1
7. int i,j,n,flag=1;
printf("enter n:");
                                                                     enter y2:0
scanf("%d",&n);
                                                                     enter x3:2
10. for(i=0;i<n;i++)
                                                                     enter y3:1
11. {
12. printf("enter x%d:",i+1);
                                                                     enter x4:3
13. scanf("%f",&x[i]);
                                                                     enter y4:10
14. printf("enter y%d:",i+1);
                                                                     newton's backward interpolation
15. scanf("%f",&y[i][0]);
                                                                     formula is applicable
16. }
17. h=x[1]-x[0];
                                                                     enter the point of interpolation:2.5
18. for(i=1;i<n-1;i++)
                                                                     difference table
19. {
20. if((x[i+1]-x[i])!=h)
21. {
                                                                       1.000 0.000
22. flag=0;
                                                                       0.000 - 1.000
23. break;
                                                                       1.000 1.000
24. }
25.}
                                                                      10.000 9.000
26. if(flag==1)
27. {
                                                                     value of the function: at x=2.500000
28. printf("newton's backward interpolation formula is applicable\n");
29. printf("enter the point of interpolation:");
                                                                     is 4.125
30. scanf("%f",&X);
31. for(j=1;j<n;j++)
32. {
                                                                     Process exited after 63.2 seconds
33. for(i=j;i<n;i++)
                                                                     with return value 0
34. y[i][j]=y[i][j-1]-y[i-1][j-1];
                                                                     Press any key to continue . . .
36. printf("difference table\n\n");
37. for(i=0;i<n;i++)
38. {
39. for(j=0;j<=1;j++)
40. printf("%7.3f",y[i][j]);
41. printf("\n");
42.}
43. u=(X-x[n-1])/h;
44. s=y[n-1][0];
45. for(i=1;i<n;i++)
46. {
47. t=t*(u+i-1)/i;
48. s=s+t*y[n-1][i];
49.}
50. printf("\nvalue of the function:");
51. printf("at x=%f is %0.3f",X,s);
52.}
53. else
54. printf("newton's backward interpolation is not applicable");
55. return 0;
```

56.}

### **Assignment 3:**

Write a program in C to find f(x) for x=0 by Lagrange's Interpolation formula using the following data:

x	-2	-1	2	4
f(x)	-9	-1	11	69

Answer: 1

Algorithm for by Lagrange's Interpolation formula:

- 1. Start
- 2. For i=0 to (n-1) do

Read xi,hi

Next i

- 3. Read x
- 4. Set s=0.0
- 5. For i=0 to (n-1) do

Set p=1.0

for j=0 to (n-1) do

If i≠j, compute p=p\*(x-xj)/(xi-xj)

Next j

s=s+p\*yi

Next i

- 6. Print s
- 7. Stop

```
Program for Lagrange's Interpolation formula:
1. /*program for lagrange's interpolation*/
```

```
#include<stdio.h>
2.
3.
   #include<math.h>
4. int main()
5. {
6. float *x,*y,X,p,s=0.0;
7. int i,j,n;
8. printf("enter n:");
9. x=(float *)malloc(n*sizeof(float));
10. y=(float *)malloc(n*sizeof(float));
11. scanf("%d",&n);
12. for(i=0;i<n;i++)
13. {
14. printf("enter x%d:",i+1);
15. scanf("%f",&x[i]);
16. printf("enter y%d:",i+1);
17. scanf("%f",&y[i]);
18.}
19. printf("enter the point of interpolation:");
20. scanf("%f",&X);
21. for(i=0;i<n;i++)
22. {
23. p=1.0;
24. for(j=0;j<n;j++)
25. {
26. if(i!=j)
27. p=p*(X-x[j])/(x[i]-x[j]);
28. }
29. s=s+p*y[i];
30.}
31. printf("value of interpolation:");
32. printf("at %f is:%0.3f",X,s);
33. return 0;
```

34. }

```
Output:
```

enter n:4 enter x1:-2 enter y1:-9 enter x2:-1 enter y2:-1 enter x3:2 enter y3:11 enter x4:4 enter y4:69

enter the point of interpolation:0 value of interpolation:at 0.000000

is:1.000

# Assignment 4:

Write a program in C by applying Trapezoidal Rule to find the value of the following definite integral:

```
\int_0^1 \frac{dx}{1+x^2}, correct the result up to 3 decimal places.
```

#### **Answer: 0.785**

```
Algorithm for Trapezoidal Rule:
```

- 1. Start
- 2. Read a,b
- 3. Read n
- 4. h=(b-a)/n
- 5. Set s = 0.0
- 6. Set i=0
- 7. s=s+f(a+(i+0)\*h)+f(a+(i+1)\*h
- 8. i=i+1
- 9. If i<n, go to step 7, else go to next step
- 10. s=s\*(h/2)
- 11. Print s
- 12. Sto**p**

#### Program for Trapezoidal Rule:

- 1. /\*TRAPEZOIDAL RULE\*/
- 2. #include<stdio.h>
- 3. #include<math.h>
- 4. /\* Define the function to be integrated here: \*/
- 5. float f(float x)
- 6. {
- 7. return 1/(1+x\*x);
- 8.
- 9. /\*Program begins\*/
- 10. main()
- 11. {
- 12. int n,i;
- 13. float a,b,h,x,sum=0,integral;
- 14. /\*Ask the user for necessary input \*/
- 15. printf("\nEnter the initial limit: ");
- 16. scanf("%f",&a);
- 17. printf("\nEnter the final limit: ");
- 18. scanf("%f",&b);
- 19. printf("\nEnter the no. of sub-intervals: ");
- 20. scanf("%d",&n);
- 21. h=fabs(b-a)/n;
- 22. for(i=1;i<n;i++)
- 23. {
- 24. x=a+i\*h;
- 25. sum=sum+f(x);
- 26.
- 27. integral=(h/2)\*(f(a)+f(b)+2\*sum);
- 28. /\*Print the answer \*/
- 29. printf("\nThe integral is: %f\n",integral);
- 30. }

Output

Enter the initial limit:

0

Enter the final limit:

1

Enter the no. of sub-intervals:

10

The integral is: 0.784981

-----

Process exited after 9.936 seconds with return value 0

Press any key to continue . . .

# **Assignment 5:**

Write a program in C by applying Simpson's 1/3 Rule to find the value of the following definite integral:

 $\int_0^1 \frac{x dx}{1+x}$ , correct the result up to 3 decimal places.

## **Answer: 0.307**

Algorithm for SIMPSON'S 1/3 RULE:

- 1. Start
- 2. Read a,b
- 3. Read n
- 4. If n%2 =0, go to next step, else go to step 13
- 5. h = (b-a)/n
- 6.set s=0.0
- 7. Set i=0
- 8. S=s+f(a+(i+0)\*h)+4\*f(a+(i+1)\*h)+f(a+(i+2)\*h)
- 9. i=i+2
- 10. If i<n, go to step 8 ,else go to next step
- 11.s=s\*(h/3)
- 12. Print s ,go to step 14
- 13. Print "Simpson's 1/3 rule is not applicable"
- 14. Stop

```
Program for SIMPSON'S 1/3 RULE:
1. /*SIMPSON'S 1/3 RULE*/
                                                          Output:
2. #include<stdio.h>
                                                          Enter the initial limit:
#include<math.h>
4. float f(float x)
                                                          Enter the final limit:
5. {
6. return x*x;
                                                          Enter the no. of sub-
7.
                                                          intervals(EVEN):
   }
8. main()
                                                          6
9. {
                                                          The integral is: 0.306830
10. int n,i;
11. float a,b,h,x,sum=0,integral;
12. printf("\nEnter the initial limit: ");
13. scanf("%f ",&a);
                                                          Process exited after 5.286
14. printf("\nEnter the final limit: ");
                                                          seconds with return value
15. scanf("%f ",&b);
                                                          0
16. printf("\nEnter the no. of sub-intervals(EVEN): ");
                                                          Press any key to continue.
17. scanf("%d",&n);
18. h=fabs(b-a)/n;
19. if(n%2==0)
20. {
21. for(i=1;i<n;i++)
22. {
23. x=a+i*h;
24. if(i%2==0)
25. {
26. sum = sum + 2*f(x);
27. }
28. else
29. {
30. sum = sum + 4*f(x);
31. }
32. }
33. integral=(h/3)*(f(a)+f(b)+sum);
34. printf("\nThe integral is: %f \n",integral);
35. }
36. else
37. printf(" error");
38. }
```

### **Assignment 6:**

29.}

```
Write a program in C by applying Weddle's Rule to find the value of the following definite integral:
\int_0^{-1} xe^x, correct the result up to 3 decimal places.
Answer: -0.2616
Algorithm for Weddle's Rule:
1.start
2. Read a,b
3.Read n
4. If n%6=0, go to next step, else go to
5. h = (b-a)/n
6. Set s=0.0
7. Set i=0
8. s = s + f(a + (i + 0) + h) + 5 + f(a + (i + 1) + h) + f(a + (i + 2) + h) + 6 + f(a + (i + 3) + h) + f(a + (i + 4) + h) + 5 + f(a + (i + 5) + h) + f(a + (i + 6) + h) + f(a +
10. If i<n,go to step 8, else go to next step
11. s=s*(3h/10)
12. Print s, go to step 14
13. Print "weddle's rule is not applicable"
14. Stop
Program for Weddle's Rule:
1. /* Weddle's Rule */
                                                                                                                                                                       Output:
2. #include<stdio.h>
3. #include<math.h>
                                                                                                                                                                       enter lower limit:0
4. #include<stdlib.h>
                                                                                                                                                                       enter upper limit:-1
5. float f(float x)
                                                                                                                                                                       enter subintervals:12
6. {
                                                                                                                                                                       the value is:0.264241
7. return (x*exp(x));
                                                                                                                                                                        _____
8. }
9. int main()
                                                                                                                                                                       Process exited after 6.345
10. {
                                                                                                                                                                       seconds with return value 21
11. float a,b,h,s=0;
                                                                                                                                                                       Press any key to continue . . .
12. int n,i;
13. printf("enter lower limit:");
14. scanf("%f",&a);
15. printf("enter upper limit:");
16. scanf("%f",&b);
17. printf("enter subintervals:");
18. scanf("%d",&n);
19. if(n%6==0)
20. {
21. h=(b-a)/n;
22. for(i=0;i< n;i=i+6)
23. s=s+f(a+(i+0)*h)+5*f(a+(i+1)*h)+f(a+(i+2)*h)+6*f(a+(i+3)*h)+f(a+(i+4)*h)+5*f(a+(i+5)*h)+f(a+(i+6)*h);
24. s=s*(3*h/10);
25. printf("the value is:%f",s);
26.}
27. else
28. printf(" rule is not applicable");
```

#### **Assignment 7:**

Write a program in C by applying Regula-Falsi method to find the value of the following algebraic equation:  $X^3 + 2x + 1 = 0$ , correct the result up to 3 decimal places.

#### Answer: -1.168

Algorithm for Regula-Falsi method:

f(x)=0,a,b,c

1. Start

2. Read a,b

3. If f(a)\*f(b)<0,go to next step,else go to step 2

4. Read e

5. x = (a \*f(b)-b\*f(a)/f(b)-f(a))

6. If f(a)\*f(b)<0,go to next step,else go to step 8

7. b=x go to step 9

8. a=x go to next step

9. If |a-b|<e, go to step 10, go to step 5

10. Print x

11. Stop

#### Program for Regula-Falsi method:

- /\*Regula Falsi mthod\*/
   #include <stdio.h>
   #include <math.h>
   float f(float x)
   {
   return (x\*x\*x-2\*x+1);
- 6. return (x\*x\*x-2\*x+1)7. }
- int main()
   {
   10.float a,b,e,x;
- 10.float a,b,e,x 11.while(1)
- 12.{
- 13.printf("enter the value of a & b:");
- 14.scanf("%f%f",&a,&b);
- 15.if(f(a)\*f(b)<0.0)
- 16.break;
- 17.printf("enter a new intervals again");
- 18.}
- 19.printf("enter error value:");
- 20.scanf("%f",&e);
- 21.do
- 22.{
- 23.x=(a\*f(b)-b\*f(a))/(f(b)-f(a));
- 24.if (f(x)\*f(a)<0.0)
- 25.b=x;
- 26.else
- 27.a=x;
- 28.}
- 29.while (fabs(a-b)>0.0);
- 30.
- 31.printf("one real root is :%0.3f",x);
- 32.}
- 33.}

# Output:

enter the value of a & b:

-1

-2

enter error value:0.001 one real root is :-1.618

-----

Process exited after 5.896 seconds with return value 0 Press any key to continue . . .

### **Assignment 8:**

Write a program in C by applying Newton-Raphson method to find the value of the following algebraic equation:

 $x^3 + 2x + 1 = 0$ , correct the result up to 3 decimal places.

#### Answer: -1.168

f(x) = 0, x0,e

8. Stop

Algorithm for Newton-Raphson method:

```
    start
    Read x0
    Read e
    x = x0
    x0= x0-f(x0)/F1(x0)
    If |x-x0|<e ,go to step 7 ,else go to step 4</li>
    Print x
```

## Program for Newton-Raphson method:

```
    /* Newton raphjson method */
    #include<stdio.h>
    #include<math.h>
```

4. float f(float z)5. {

6. return (z\*z\*z-2\*z+1);7. }

8. float f1(float z)

9. {
 10. return (3\*z\*z-2);
 11. }

12. int main ()

13. {14. float x0,x,e;

15. printf(" enter the initial guess of root : ");

16. scanf(" %f",&x0);

17. printf(" enter the error value:");

18. scanf("%f",&e);

19. do

20. { 21. x=x0;

22. x0=x0-f(x0)/f1(x0);

23. }

24. while (fabs(x-x0)>e);

25. printf("one real root of the equation is :%0.3f",x);

26. }

# Output:

enter the initial guess of root : -2

enter the error value:0.001 one real root of the equation is :-1.618

-----

Process exited after 9.981 seconds with return value 0 Press any key to continue . . .

# **Assignment 9:**

Write a program in C by applying Euler's method to find the solution of the following differential equation:

 $\frac{dx}{dy} = \frac{x-y}{x+y}$ , taking y=1 at x=0, find y at x=0.1, taking step length 0.02 and print the

result correct up to 3 decimal places. Answer: 1.093

```
Algorithm for eular method:
```

```
1.start
```

- 2. Read x0,y0
- 3. Read xn
- 4. Read h
- 5. y0 = y0 + h\*f(x0,y0)
- 6. x0=x0+h
- 7. If x0<xn, go to step 5, go to next step
- 8. Print y0
- 9. Stop

Program for eular method:

- /\*eular method\*/
- 2. #include<stdio.h>
- 3. #include<math.h>
- 4. float f(float x,float y)
- 5. {
- 6. return (y-x)/(y+x);
- 7. }
- 8. int main()
- 9. {
- 10. float x0,y0,xn,h;
- 11. printf(" enter the value of x0,y0 & xn:\n");
- 12. scanf("%f%f%f",&x0,&y0,&xn);
- 13. printf("enter step length:\n");
- 14. scanf("%f",&h);
- 15. do
- 16. {
- 17. y0=y0+h\*f(x0,y0);
- 18. x0=x0+h;
- 19.}
- 20. while(x0 < xn);
- 21. printf("the value of y0 is:%f",y0);
- 22. }

Output:

enter the value of x0,y0 & xn:

0

1

0.1

enter step length:

.02

the value of y0 is:1.109478

-----

Process exited after 7.022 seconds with

return value 0

Press any key to continue . . .

### Assignment 10:

Write a program in C by applying R-K method of order 4 to find the solution of the following differential equation:  $=\frac{x-y}{x+y}$ , taking y=1 at x=0, find y at x=0.1, taking step length 0.02 and print the result correct up to 3 decimal places. Answer: 1.092

### Algorithm for R-K method of order 4: 1.start 2.Read x0,y0,xn,h 3.k1=h\*f(x0,y0)4.k2=h\*f(x0+h/2,y0+k1/2)5.k3=h\*f(x0+h/2,y0+k2/2)6.k4=h\*f(x0+h,y0+k3)7.y0=y0+(k1+2\*(k2+k3)+ke)/68.x0=x0+h9.If x0<=xn,go to step 3,go to next step

### Program for R-K method of order 4:

- /\*program for rk method\*/ 2. #include<stdio.h>
- 3. #include<math.h>
- 4. float f(float,float);
- 5. int main()
- 6. {

10.print y0 11.stop

- 7. float x0,y0,xn,k1,k2,k3,k4,h;
- 8. printf("enter x0:");
- 9. scanf("%f",&x0);
- 10. printf("enter y0:");
- 11. scanf("%f",&y0);
- 12. printf("enter xn:");
- 13. scanf("%f",&xn);
- 14. printf("enter step length:");
- 15. scanf("%f",&h);
- 16. do
- 17. {
- 18. k1=h\*f(x0,y0);
- 19. k2=h\*f((x0+h/2),(y0+k1/2));
- 20. k3=h\*f((x0+h/2),(y0+k2/2));
- 21. k4=h\*f(x0+h,y0+k3);
- 22. y0=y0+(k1+2\*(k2+k3)+k4)/6;
- 23. x0=x0+h;
- 24. }
- 25. while(x0<xn);
- 26. printf("the solution is:%0.3f",y0);
- 27. return 0;
- 28. }
- 29. float f(float x,float y)
- 30. {
- 31. return (y-x)/(y+x);
- 32. }

## **Output:**

enter x0:0

enter y0:1

enter step length:0.02

enter xn:0.1

the solution is:1.109

# **Assignment 11:**

Write a program in C using Gauss-Seidel iterative method to solve the following system of linear equations:

$$3x + y + 5z = 13$$
  
 $5x - 2y + z = 4$   
 $x + 6y - 2z = -1$ 

13. Stop

Answer: x=0.55, y=0.47, z=2.18 Algorithm for Gauss-Seidel iterative method:

```
x + 6y - 2z = -1
AX = b, Ab
1.start
2. Read n
3. For i = 0 to (n-1) do
     For j=0 to n do
Read aij
Next j
Next i
4. Read count
5. Set all xi=0.0
            n-1
6. If |aii| > \Sigma |aij|, go to next
            i,j=0 step, else
            i≠j. go to step 12
7. Set k=1
8. For i=0 to (n-1) do
Set xi = ai,n
For j=0 to (n-1) do
If i≠j; compute xi= xi - aij*xj
Next i
xi= xi / aii
Next i
9. If k<=count, go to step 8, else go to next step
10. For i=0 to (n-1) do
Print xi
Next i
11. Go to step 13
12. Print ("gauss Seidel method cannot be applicable")
```

```
77. }
Program for Gauss-Seidel iterative method: 39. flag=1;
1. /* program for Gauss-Seidel method */ 40. for(j=0;j<n+1;j++)
                                                                                            78. x[i]=x[i]/A[i][i];
2. #include<stdio.h>
                                                                                            79. }
                                              42. A[pos][j]=a[i][j];
   #include<math.h>
                                                                                            80. k++;
4. int main()
                                              43. }
                                                                                            81. }
                                              44. }
                                                                                            82. printf("Solutions: \n");
5. {
6. float
                                              45. else
                                                                                            83. for(i=0;i<n;i++)
    a[10][10],A[10][10],s,max,x[10]={0.0};
                                              46. {
                                                                                            84. {
7. int i,j,n,k,flag=0,pos,cnt;
                                                                                            85. printf("x%d=%0.3f\n",i+1,x[i]);
                                              47. flag=0;
printf("Enter n:");
                                              48. break;
                                                                                            86. }
9. scanf("%d",&n);
                                              49. }
                                                                                            87. }
10. printf("Enter Augmented matrix: \n");
                                              50. }
                                                                                            88. else
11. for(i=0;i<n;i++)
                                              51. if(flag==1)
                                                                                            89. {
12. {
                                                                                            90. printf("System of equations are not
                                              52. {
13. for(j=0;j< n+1;j++)
                                                                                                diagonally dominant\n");
                                              53. printf("System of equations are
14. {
                                                  diagonally dominant\n");
                                                                                            91. printf("Gauss-Seidel Method cannot be
                                                                                                applicable");
                                              54. printf("The augmented matrix after
15. scanf("%f",&a[i][j]);
                                                  rearrangement of equations :\n");
                                                                                            92. }
16. }
                                              55. for(i=0;i<n;i++)
                                                                                            93. return 0;
17. }
                                              56. {
                                                                                            94. }
18. for(i=0;i<n;i++)
                                              57. for(j=0;j<n;j++)
                                                                                               Output:
19. {
                                              58. {
                                                                                               Enter n:3
20. max=a[i][0];
                                              59. printf("%7.3f",A[i][j]);
                                                                                               Enter Augmented matrix:
21. for(j=0;j<n;j++)
                                              60. }
                                                                                               3 1 5 13
22. {
                                              61. printf("\n");
                                                                                               5 -2 1 4
23. if(a[i][j] >= max)
                                              62. }
                                                                                               16-2-1
24. {
                                              63. printf("Enter no. iterations: ");
                                                                                               System of equations are diagonally
25. max=a[i][j];
                                                                                               dominant
                                              64. scanf("%d",&cnt);
26. pos=j;
                                              65. k=0;
                                                                                               The augmented matrix after
27. }
                                                                                               rearrangement of equations:
                                              66. while(k<cnt)
28. }
                                                                                                5.000 -2.000 1.000
                                              67. {
29. s=0.0;
                                                                                                1.000 6.000 -2.000
                                              68. for(i=0;i<n;i++)
30. for(j=0;j<n;j++)
                                                                                                3.000 1.000 5.000
                                              69. {
31. {
                                                                                               Enter no. iterations: 25
                                              70. x[i]=A[i][j];
32. if(j!=pos)
                                                                                               Solutions:
                                              71. for(j=0;j< n;j++)
33. {
                                                                                               x1=0.552
                                              72. {
34. s=s+fabs(a[i][j]);
                                              73. if(i!=j)
                                                                                               x2=0.467
35. }
                                              74. {
                                                                                               x3=2.176
36. }
                                              75. x[i]=x[i]-A[i][j]*x[j];
37. if(fabs(a[i][pos])>s)
                                              76. }
38. {
```

Assignment 12:		17.	for(j=0;j <n+1;j++)< td=""></n+1;j++)<>		
Write a program in C using Gauss-Elimination method to solve the following system of linear equations:		18.	scanf("%f",&a[i][j]);		
5x1 - x2 + x3 = 10		19.	/elementary transformation/		
2x1 + 4x2 = 12		20.	for(k=0;k <n-1;k++)< td=""></n-1;k++)<>		
	? + 5x3 = -1	21.	{		
	r: <i>x1</i> =2.556, <i>x2</i> =1.772, <i>x3</i> =-1.056	22.	for(i=k+1;i <n;i++)< td=""></n;i++)<>		
	thm for Gauss-Elimination method:	23.	{		
1.start	•	24.	r=a[i][k]/a[k][k];		
2. Read		25.	for(j=0;j <n+1;j++)< td=""></n+1;j++)<>		
	=0 to n do	26.	a[i][j]=a[i][j]-r*a[k][j];		
		27.	}		
-	to (n+1) do	28.	}		
Read a	IJ	29.	/printing of transformed augmented matrix/		
Next i		30.	for(i=0;i <n;i++)< td=""></n;i++)<>		
Next j		31.	{		
	x=0 to (n-1) do	32.	for(j=0;j <n+1;j++)< td=""></n+1;j++)<>		
-	(+1) to n do	33.	printf("%7.3f",a[i][j]);		
	te ,r = a[i][j]/a[k][k]	34.	printf("\n");		
-	to ( n+1) do	35.	}		
a[i][j]=a[i][j]-r*a[k][j]		36.	/back substitution/		
Next j		37.	for(i=n-1;i>=0;i)		
Next k		38.	{		
	= (n-1) to 0	39.	x[i]=a[i][n];		
x[i]=a[i		40.	for(j=i+1;j <n;j++)< td=""></n;j++)<>		
for j=(i-	+1) to n	41.	x[i]=x[i]-a[i][j]*x[j];		
x[i]=x[i]-a[i][j]*x[j]		42.	x[i]=x[i]/a[i][i];		
Next j		43.	}		
x[i]=x[i]/a[i][j]		44.	/printing of solutions/		
Next i		45.	printf("the solutions are:\n");		
6. Print the result x[i]		46.	for(i=0;i <n;i++)< td=""></n;i++)<>		
7. Stop		47.	printf("x%d=%0.4f\n",i+1,x[i]);		
Progra	m for Gauss-Elimination method <u>:</u>	48.	return 0;		
1.	program for gauss elimination method/	49.	}		
2.	#include <stdio.h></stdio.h>	.5.	,		
3.	#include <stdlib.h></stdlib.h>	• 0	utput:		
4.	#include <math.h></math.h>	enter			
5.	int main()	enter the augmented matrix:			
6.	{	5 -1 1 10			
7.	float **a,*x,r;	2 4 0 12			
8.	int n,i,j,k;	1 1 5 -1			
9.	printf("enter n:");		) -1.000		
10.	scanf("%d",&n);	0.000 4.400 -0.400 8.000			
11.	x=(float *)malloc(n*sizeof(float));	0.000 0.000 4.909 -5.182			
12.	<pre>12. a=(float **)malloc(n*sizeof(float));</pre>		the solutions are:		
13.	13. for(i=0;i <n+1;i++)< td=""><td colspan="3"></td></n+1;i++)<>				
14.	14. a[i]=(float *)malloc(n*sizeof(float));		x1=2.5556		
15.	15. printf("enter the augmented matrix:\n");		x2=1.7222 x3=-1.0556		
16.	for(i=0;i <n;i++)< td=""><td>x3=-1.</td><td>υσου</td></n;i++)<>	x3=-1.	υσου		