Aim: To find the roots of linear equations by Gauss-Seidel Iterative Method

Algorithm:

1. Enter the no. of equations, n.
2. Create an nx(n+1) matrix, which will be the augmented matrix.
3. Create an array 'x' of size 'n' which will store the solutions.
4. Enter the elements of augmented matrix.
5. Enter the initial guesses.
6. Enter the tolerance limit,eps.
7. Pivoting:

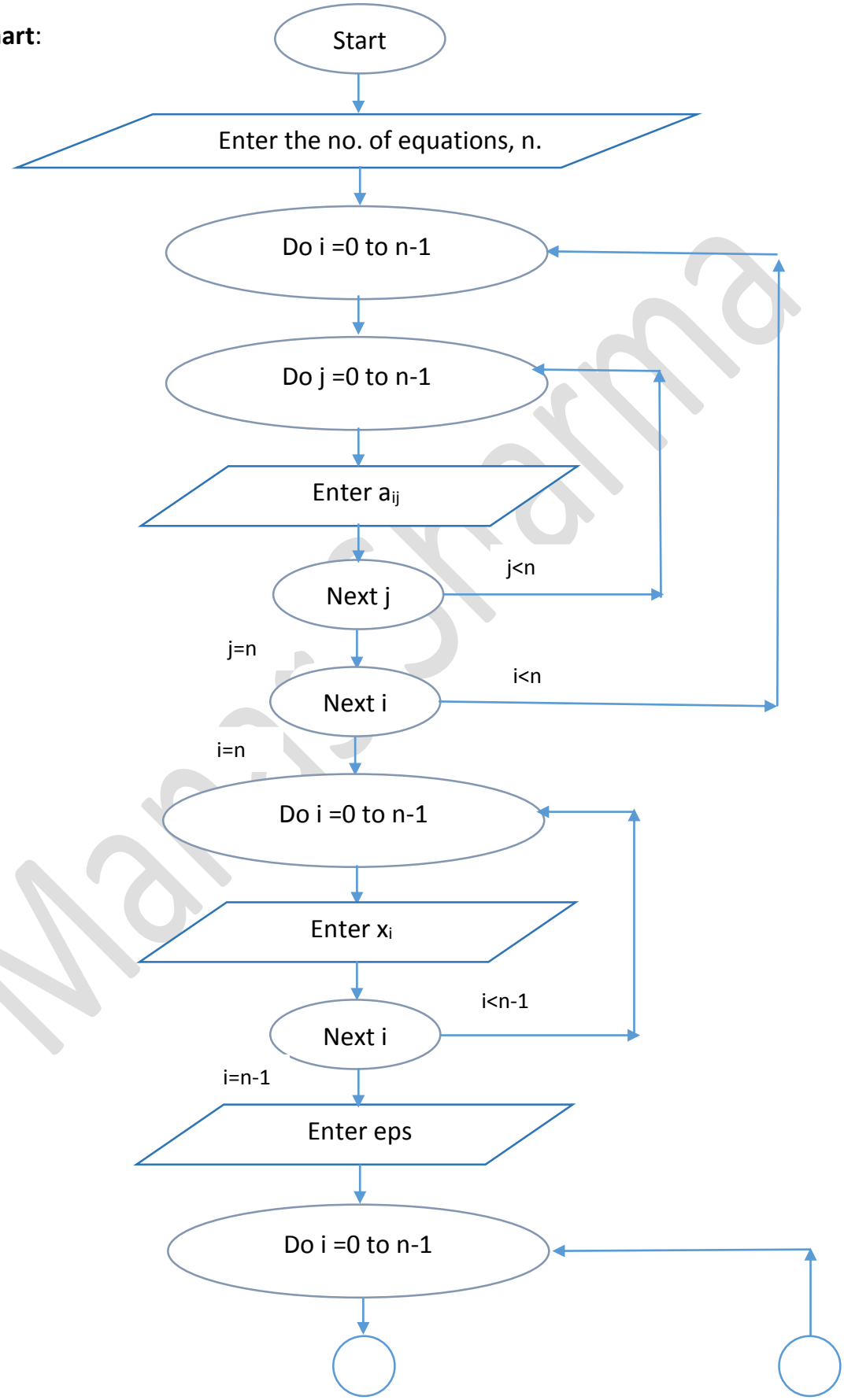
```
For i=0 to n-1
    For k=i+1 to n-1
        If a_ii<a_ki
          Then For j=0 to n
                Swap a_ij with a_kj
```

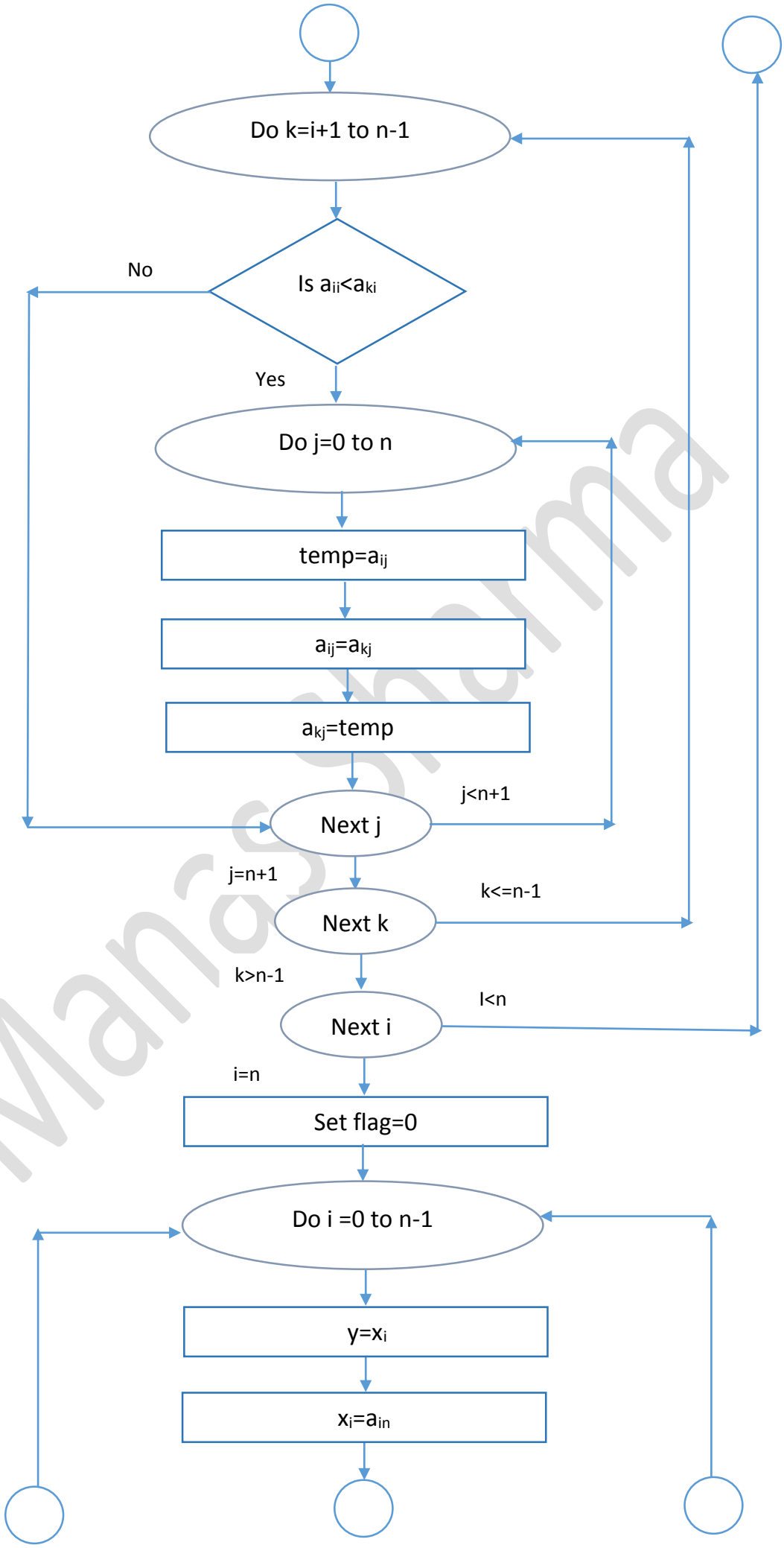
8. Set flag=0.
9. Do:

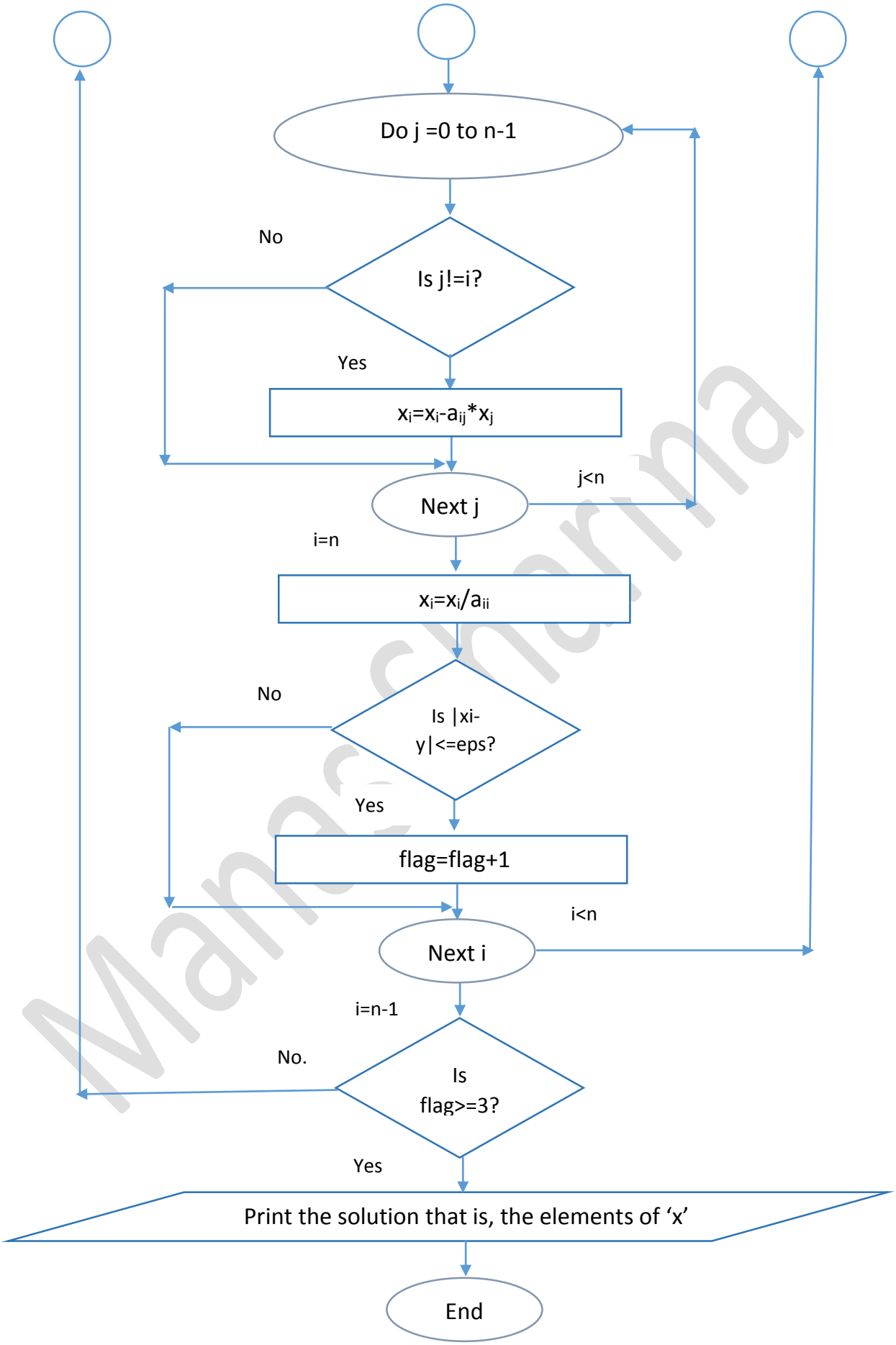
```
Do For i=0 to n-1

    y=x_i
    x_i=a_in
    Do For j=0 to n-1
        If j!=i
        Then x_i=x_i-a_ij*x_j
    End For j.
    x_i=x_i/a_ii
    If |xi-y|<=eps
    Then increment flag.
End For i.

Until flag<3
```

10. Print the solution that is, the elements of 'x' array.

Flow Chart:







Program:

```
//Gaus-seidel
#include<iostream>
#include<iomanip>
#include<cmath>
using namespace std;
int main()
{
    cout.precision(4);
    cout.setf(ios::fixed);
    int n,i,j,k,flag=0,count=0;
    cout<<"\nEnter the no. of equations\n";
    cin>>n;                     //Input no. of equations
    double a[n][n+1];             //declare a 2d array for
storing the elements of the augmented matrix
    double x[n];                 //declare an array to store
the values of variables
    double eps,y;
    cout<<"\nEnter the elements of the augmented matrix row-
wise:\n";
    for (i=0;i<n;i++)
        for (j=0;j<=n;j++)
            cin>>a[i][j];
    cout<<"\nEnter the initial values of the variables:\n";
    for (i=0;i<n;i++)
        cin>>x[i];
    cout<<"\nEnter the accuracy upto which you want the
solution:\n";
    cin>>eps;
    for (i=0;i<n;i++)
//Pivotisation(partial) to make the equations diagonally
dominant
        for (k=i+1;k<n;k++)
            if (a[i][i]<a[k][i])
                for (j=0;j<=n;j++)
                {
                    double temp=a[i][j];
                    a[i][j]=a[k][j];
                    a[k][j]=temp;
                }
    cout<<"Iter"<<setw(10);
    for(i=0;i<n;i++)
        cout<<"x"<<i<<setw(18);
    cout<<"\n-----------------------------------------------------
---------------------";
    do                            //Perform iterations to
calculate x1,x2,...xn
    {
        cout<<"\n"<<count+1<<"."<<setw(16);
```

```
        for (i=0;i<n;i++)                 //Loop that
calculates x1,x2,...xn
        {
            y=x[i];
            x[i]=a[i][n];
            for (j=0;j<n;j++)
            {
                if (j!=i)
                x[i]=x[i]-a[i][j]*x[j];
            }
            x[i]=x[i]/a[i][i];
            if (abs(x[i]-y)<=eps)            //Compare the ne
value with the last value
                flag++;
            cout<<x[i]<<setw(18);
        }
        cout<<"\n";
        count++;
    }while(flag<3);                        //If the values of
all the variables don't differ from their previous values with
error more than eps then flag must be 3 and hence stop the
loop

    cout<<"\n The solution is as follows:\n";
    for (i=0;i<n;i++)
        cout<<"x"<<i<<" = "<<x[i]<<endl;        //Print the
contents of x[]
    return 0;
}
```

Output:

```
Enter the no. of equations
3

Enter the elements of the augmented matrix row-wise:
6       -2      1       11
1       2       -5      -1
-2      7       2       5

Enter the initial values of the variables:
0       0       0

Enter the accuracy upto which you want the solution:
0.0001
Iter          x0                  x1                  x2
-----------------------------------------------------------------
1.          1.8333              1.2381              1.0619

2.          2.0690              1.0020              1.0146

3.          1.9982              0.9953              0.9978

4.          1.9988              1.0003              0.9999

5.          2.0001              1.0001              1.0001

6.          2.0000              1.0000              1.0000

7.          2.0000              1.0000              1.0000

 The solution is as follows:
x0 = 2.0000
x1 = 1.0000
x2 = 1.0000
```