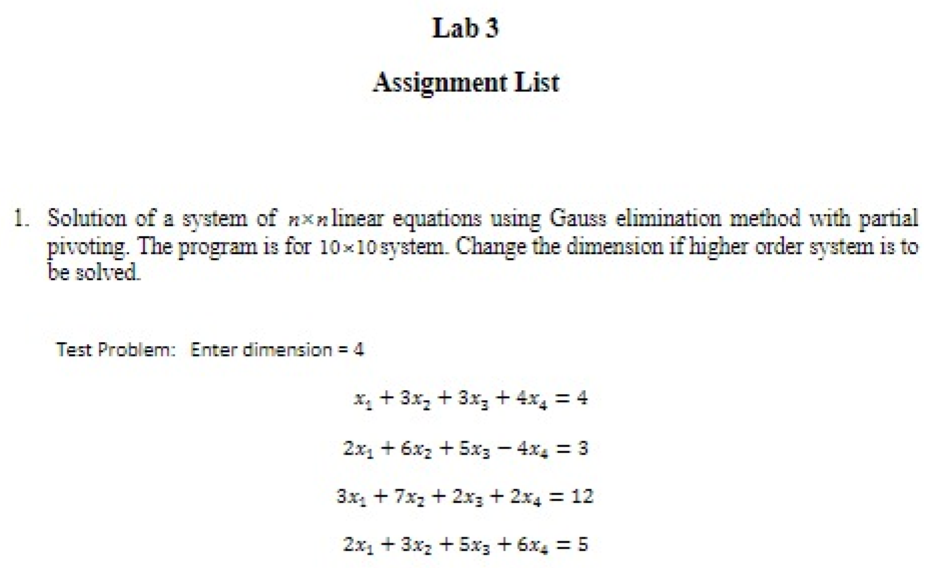
NM LAB ASSIGNMENT-Lab 3



Lab 3

Assignment List

1. Solution of a system of $n \times n$ linear equations using Gauss elimination method with partial pivoting. The program is for 10×10 system. Change the dimension if higher order system is to be solved.

Test Problem: Enter dimension = 4

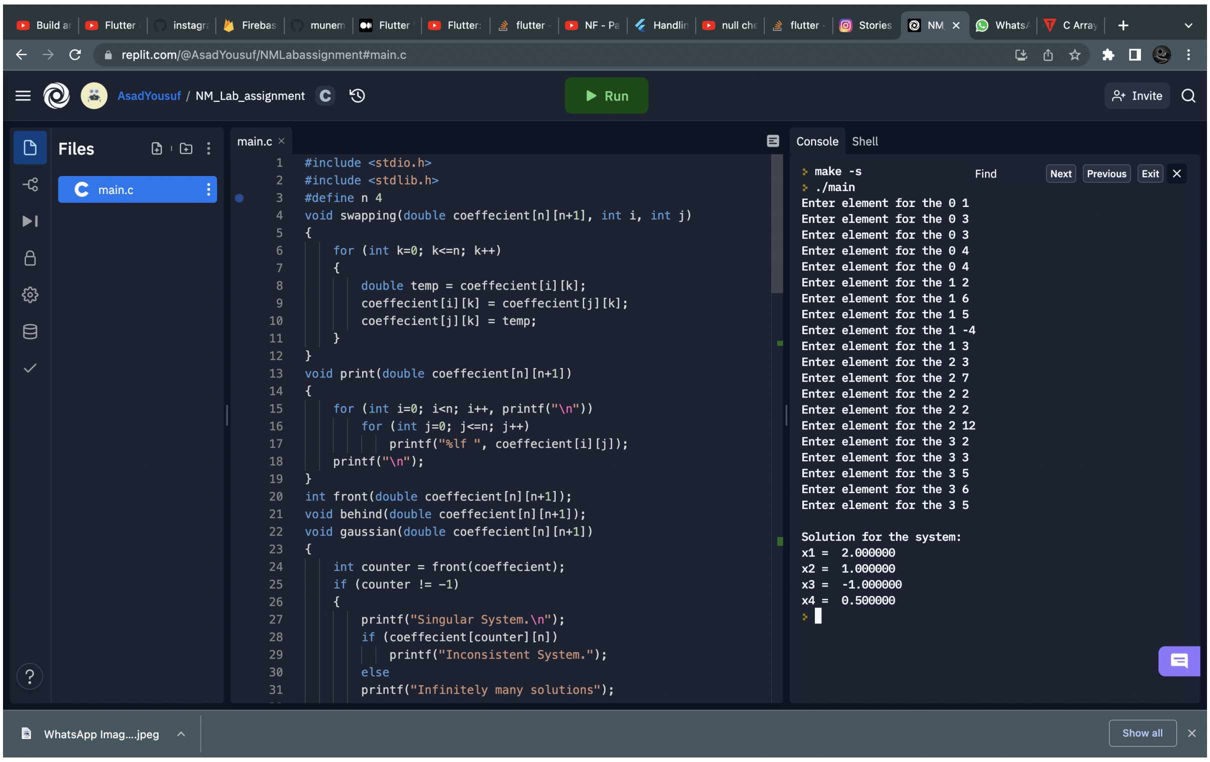
$$x_1 + 3x_2 + 3x_3 + 4x_4 = 4$$

$$2x_1 + 6x_2 + 5x_3 - 4x_4 = 3$$

$$3x_1 + 7x_2 + 2x_3 + 2x_4 = 12$$

$$2x_1 + 3x_2 + 5x_3 + 6x_4 = 5$$

```
#include <stdio.h>
#include <stdlib.h>
#define n 4
void swapping(double coeffecient[n][n+1], int i, int j)
{
    for (int k=0; k<=n; k++)
    {
        double temp = coeffecient[i][k];
        coeffecient[i][k] = coeffecient[j][k];
        coeffecient[j][k] = temp;
    }
}
void print(double coeffecient[n][n+1])
{
    for (int i=0; i<n; i++, printf("\n"))
        for (int j=0; j<=n; j++)
            printf("%lf ", coeffecient[i][j]);
    printf("\n");
}
int front(double coeffecient[n][n+1]);
void behind(double coeffecient[n][n+1]);
void gaussian(double coeffecient[n][n+1])
{
    int counter = front(coeffecient);
    if (counter != -1)
    {
        printf("Singular System.\n");
        if (coeffecient[counter][n])
            printf("Inconsistent System.");
        else
        printf("Infinitely many solutions");
        return;
    }
    behind(coeffecient);
}
int front(double coeffecient[n][n+1])
{
    for (int k=0; k<n; k++)
    {
        int i_max = k;
        int v_max = coeffecient[i_max][k];
        for (int i = k+1; i < n; i++)
            if (abs(coeffecient[i][k]) > v_max)
                v_max = coeffecient[i][k], i_max = i;
        if (!coeffecient[k][i_max])
            return k;
        if (i_max != k)
            swapping(coeffecient, k, i_max);
        for (int i=k+1; i<n; i++)
        {
            double f = coeffecient[i][k]/coeffecient[k][k];
            for (int j=k+1; j<=n; j++)
                coeffecient[i][j] -= coeffecient[k][j]*f;
            coeffecient[i][k] = 0;
        }
    }
    return -1;
}
void behind(double coeffecient[n][n+1])
{
    double x[n];
    for (int i = n-1; i >= 0; i--)
    {
        x[i] = coeffecient[i][n];
        for (int j=i+1; j<n; j++)
        {
            x[i] -= coeffecient[i][j]*x[j];
        }
        x[i] = x[i]/coeffecient[i][i];
    }
    printf("\nSolution for the system:\n");
    for (int i=0; i<n; i++)
        printf("x%d =  %lf\n",i+1, x[i]);
}
int main()
{
    double coeffecient[n][n+1];
    for(int i=0;i<n;i++)
    {
      for(int j=0;j<n+1;j++)
      {
        printf("Enter element for the %d ",i,"th equation");
       scanf("%d",&coeffecient[i][j]);
      }
    }
    gaussian(coeffecient);
    return 0;
}
```



```c
#include <stdio.h>
#include <stdlib.h>
#define n 4
void swapping(double coeffecient[n][n+1], int i, int j)
{
    for (int k=0; k<=n; k++)
    {
        double temp = coeffecient[i][k];
        coeffecient[i][k] = coeffecient[j][k];
        coeffecient[j][k] = temp;
    }
}
void print(double coeffecient[n][n+1])
{
    for (int i=0; i<n; i++, printf("\n"))
        for (int j=0; j<=n; j++)
            printf("%lf ", coeffecient[i][j]);
    printf("\n");
}
int front(double coeffecient[n][n+1]);
void behind(double coeffecient[n][n+1]);
void gaussian(double coeffecient[n][n+1])
{
    int counter = front(coeffecient);
    if (counter != -1)
    {
        printf("Singular System.\n");
        if (coeffecient[counter][n])
            printf("Inconsistent System.");
        else
        printf("Infinitely many solutions");
```

```
> make -s
> ./main
Enter element for the 0 1
Enter element for the 0 3
Enter element for the 0 3
Enter element for the 0 4
Enter element for the 0 4
Enter element for the 1 2
Enter element for the 1 6
Enter element for the 1 5
Enter element for the 1 -4
Enter element for the 1 3
Enter element for the 2 3
Enter element for the 2 7
Enter element for the 2 2
Enter element for the 2 2
Enter element for the 2 12
Enter element for the 3 2
Enter element for the 3 3
Enter element for the 3 5
Enter element for the 3 6
Enter element for the 3 5

Solution for the system:
x1 =  2.000000
x2 =  1.000000
x3 =  -1.000000
x4 =  0.500000
>
```