# 7.1 高斯消元法

**7.1.1.概述**

高斯消元法(Gaussian elimination)是求解线性方阵组的一种算法，它也可用来求矩阵的秩，以及求可逆方阵的逆矩阵。它通过逐步消除未知数来将原始线性系统转化为另一个更简单的等价的系统。它的实质是通过初等行变化(Elementary row operations)，将线性方程组的增广矩阵转化为行阶梯矩阵(row echelon form).

**7.1.2.模板**

**以洛谷P3389 高斯消元法为例**

**题目描述**

给定一个线性方程组，对其求解

**输入格式**

第一行一个正整数n

第二至n+1行，每行n+1个整数，为a1,a2……和b，代表一组方程

**输出格式**

共n行，每行一个数，第i行为xi（保留2位小数）

如果不存在唯一解，在第一行输出“No Solution”.

**输入样例**

3

1 3 4 5

1 4 7 3

9 3 2 2

**输出样例**

-0.97

5.18

-2.39

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. #define int long long
4. **const** **int** N=105;
5. **double** mp[N][N];
6. **int** n;
7. **signed** main()
8. {
9. cin>>n;
10. **for**(**int** i=1;i<=n;i++)
11. {
12. **for**(**int** j=1;j<=n+1;j++)
13. {
14. cin>>mp[i][j];
15. }
16. }
17. **for**(**int** i=1;i<=n;i++)//列主元
18. {
19. **double** maxx=mp[i][i];//寻找一列中最大的元素，将此行放在最前面，防止主对角线为mp[i][i]为0
20. **int** pos=i;
21. **for**(**int** j=i+1;j<=n;j++)
22. {
23. **if**(mp[j][i]>maxx)
24. {
25. pos=j;
26. maxx=mp[j][i];
27. }
28. }
29. **if**(pos!=i)
30. {
31. **for**(**int** j=i;j<=n+1;j++)
32. {
33. **double** k=mp[pos][j];
34. mp[pos][j]=mp[i][j];
35. mp[i][j]=k;
36. }
37. }
38. **for**(**int** j=i+1;j<=n;j++)//将第i列的元素置0
39. {
40. **double** tmp=1.0\*mp[j][i]/mp[i][i];
41. **for**(**int** k=i;k<=n+1;k++)
42. {
43. mp[j][k]-=tmp\*mp[i][k];
44. }
45. }
46. }
47. **bool** flag=**true**;
48. **for**(**int** i=n;i>=1;i--)
49. {
50. **for**(**int** j=i+1;j<=n;j++)
51. {
52. mp[i][n+1]-=mp[i][j]\*mp[j][n+1];//mp[j][n+1]存储的是xj的值
53. }
54. **if**(mp[i][i]==0)//当主对角线有元素为0时无解
55. {
56. cout<<"No Solution"<<endl;
57. flag=**false**;
58. **break**;
59. }
60. mp[i][n+1]/=mp[i][i];//主对角线不一定为1，将结果进行处理
61. }
62. **if**(flag)
63. **for**(**int** i=1;i<=n;i++)
64. {
65. printf("%.2lf\n",mp[i][n+1]);
66. }
67. **return** 0;
68. }

**7.1.3.经典例题**

**7.1.3.1 洛谷P2455 线性方程组**

**题目描述**

已知n元线性方程组，求解方程组

**输入格式**

第一行输入未知数个数n

接下来n行，每行n+1个整数，表示每一个方程的系数及方程右边的值

**输出格式**

如果有唯一解，则输出解，保留两位小数

无解输出-1

无穷多解输出0

**输入样例**

3

2 -1 1 1

4 1 -1 5

1 1 1 0

**输出样例**

x1=1.00

x2=0.00

x3=-1.00

**思路：本题比模板题多了判断无解与无穷解的情况**

**题解：**

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. #define int long long
4. **const** **double** eps=1e-8;
5. **int** n;
6. **const** **int** N=55;
7. **double** mp[N][N];
8. **bool** flag=**true**;
9. **void** Gauss(**double** mp[][N])
10. {
11. **for**(**int** i=1;i<=n;i++)
12. {
13. **double** maxx=mp[i][i];
14. **int** pos=i;
15. **for**(**int** j=i+1;j<=n;j++)
16. {
17. **if**(mp[j][i]-maxx>eps)
18. {
19. maxx=mp[j][i];
20. pos=j;
21. }
22. }
23. **if**(fabs(maxx)<=eps) **continue**;//若最大值为0，说明mp[i][i]整列为0，进入下一列
24. **if**(pos!=i) **for**(**int** j=i;j<=n+1;j++) swap(mp[pos][j],mp[i][j]);
25. **for**(**int** j=i+1;j<=n;j++)
26. {
27. **if**(fabs(mp[j][i])<=eps) **continue**;
28. **double** tmp=mp[j][i]/mp[i][i];
29. **for**(**int** k=i;k<=n+1;k++)
30. {
31. mp[j][k]-=tmp\*mp[i][k];
32. }
33. }
34. }
35. **for**(**int** i=1;i<=n;i++)//判断0,0,0,0……x 的情况
36. {
37. **if**(fabs(mp[i][n+1])>eps)
38. {
39. **bool** check0=**false**;
40. **for**(**int** j=1;j<=n;j++) **if**(fabs(mp[i][j])>eps)
41. {
42. check0=**true**;
43. **break**;
44. }
45. **if**(!check0)
46. {
47. flag=**false**;
48. cout<<"-1"<<endl;
49. **return** ;
50. }
51. }
52. }
53. flag=**true**;
54. **for**(**int** i=n;i>=1;i--)
55. {
56. **for**(**int** j=i+1;j<=n;j++)
57. {
58. mp[i][n+1]-=mp[i][j]\*mp[j][n+1];
59. }
60. **if**(fabs(mp[i][i])<=eps&&fabs(mp[i][n+1])>eps){cout<<"-1"<<endl; flag=**false**; **return** ;};//首先判断无解，再判断无穷多解
61. **if**(fabs(mp[i][i]<=eps&&fabs(mp[i][n+1])<=eps)) {cout<<"0"<<endl; flag=**false**; **return** ;}
62. mp[i][n+1]/=mp[i][i];
63. }
64. }
65. **signed** main()
66. {
67. cin>>n;
68. **for**(**int** i=1;i<=n;i++)
69. {
70. **for**(**int** j=1;j<=n+1;j++)
71. {
72. cin>>mp[i][j];
73. }
74. }
75. Gauss(mp);
76. **if**(flag)
77. **for**(**int** i=1;i<=n;i++) printf("x%d=%.2lf\n",i,mp[i][n+1]+eps);
78. **return** 0;
79. }

**7.1.3.2 acwing 883.** **高斯消元解线性方程组**

****

**代码**

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. **const** **int** N=105;
4. **const** **double** eps=1e-8;
5. **double** a[N][N];
6. **void** out(**int** n)
7. {
8. **for**(**int** i=1;i<=n;i++)
9. {
10. **for**(**int** j=1;j<=n+1;j++)
11. {
12. cout<<a[i][j]<<" ";
13. }
14. cout<<endl;
15. }
16. cout<<endl;
17. }
18. **void** guass(**int** n)
19. {
20. **int** r,c;
21. **for**(r=1,c=1;c<=n;c++)
22. {
23. **int** pos=r;
24. **for**(**int** i=r+1;i<=n;i++)
25. {
26. **if**(fabs(a[pos][c])<fabs(a[i][c])) pos=i;
27. }
28. **if**(fabs(a[pos][c])<eps) **continue**;
29. **for**(**int** i=c;i<=n+1;i++) swap(a[pos][i],a[r][i]);
30. **for**(**int** i=n+1;i>=c;i--) a[r][i]/=a[r][c];
31. **for**(**int** i=r+1;i<=n;i++)
32. {
33. **if**(fabs(a[i][c])>eps)
34. {
35. **for**(**int** j=n+1;j>=c;j--)
36. {
37. a[i][j]-=a[i][c]\*a[r][j];
38. }
39. }
40. }
41. r++;
42. }
43. **if**(r<=n)
44. {
45. **for**(**int** i=r;i<=n;i++)
46. {
47. **if**(fabs(a[i][n+1])>eps)
48. {
49. cout<<"No solution"<<endl;
50. **return** ;
51. }
52. }
53. cout<<"Infinite group solutions"<<endl;
54. **return** ;
55. }
56. //out(n);
57. **for**(**int** i=n;i>=1;i--)
58. {
59. **for**(**int** j=i+1;j<=n;j++)
60. {
61. a[i][n+1]-=a[i][j]\*a[j][n+1];
62. }
63. }
64. **for**(**int** i=1;i<=n;i++) printf("%.2lf\n",a[i][n+1]);
65. }
66. **int** main()
67. {
68. **int** n;
69. cin>>n;
70. **for**(**int** i=1;i<=n;i++)
71. {
72. **for**(**int** j=1;j<=n+1;j++)
73. {
74. cin>>a[i][j];
75. }
76. }
77. guass(n);
78. **return** 0;
79. }

**7.1.3.3 acwing 884.** **高斯消元解异或线性方程组**

****

**代码**

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. **int** n;
4. **const** **int** N=105;
5. **int** a[N][N];
6. **void** out(**int** n)
7. {
8. **for**(**int** i=1;i<=n;i++)
9. {
10. **for**(**int** j=1;j<=n+1;j++)
11. {
12. cout<<a[i][j]<<" ";
13. }
14. cout<<endl;
15. }
16. cout<<endl;
17. }
18. **void** guass(**int** n)
19. {
20. **int** c,r;
21. **for**(c=1,r=1;c<=n;c++)
22. {
23. **int** pos=r;
24. **for**(**int** i=r;i<=n;i++)
25. {
26. **if**(a[i][c])
27. {
28. pos=i;
29. **break**;
30. }
31. }
32. **if**(!a[pos][c]) **continue**;
33. **for**(**int** i=c;i<=n+1;i++) swap(a[pos][i],a[r][i]);
34. **for**(**int** i=r+1;i<=n;i++)
35. {
36. **if**(a[i][c])
37. {
38. **for**(**int** j=n+1;j>=c;j--)
39. {
40. a[i][j]^=a[r][j];
41. }
42. }
43. }
44. r++;
45. }
46. //out(n);
47. **if**(r<=n)
48. {
49. **for**(**int** i=r;i<=n;i++)
50. {
51. **if**(a[i][n+1])
52. {
53. cout<<"No solution"<<endl;
54. **return** ;
55. }
56. }
57. cout<<"Multiple sets of solutions"<<endl;
58. **return** ;
59. }
60. **for**(**int** i=n;i>=1;i--)
61. {
62. **for**(**int** j=i+1;j<=n;j++)
63. {
64. a[i][n+1]^=a[i][j]\*a[j][n+1];
65. }
66. }
67. **for**(**int** i=1;i<=n;i++) cout<<a[i][n+1]<<endl;
68. }
69. **int** main()
70. {
71. cin>>n;
72. **for**(**int** i=1;i<=n;i++)
73. **for**(**int** j=1;j<=n+1;j++)
74. cin>>a[i][j];
75. guass(n);
76. **return** 0;
77. }