(14%) Matrix Determinant

Description

The determinant of an nxn square matrix M=[m_i,j] can be calculated by expanding along any row j, 1 ≤ j ≤ n, as follows:

$$|\mathrm{M}| = \sum_{i=1}^{n} m_{i,j}(-1)^{i+j} M_{ij}, \qquad M_{ij} = \begin{vmatrix} m_{1,1} & \cdots & m_{1,j-1} & m_{1,j+1} & \cdots & m_{1,n} \\ \vdots & & \vdots & \vdots & & \vdots \\ m_{i-1,1} & \cdots & m_{i-1,j-1} & m_{i-1,j+1} & \cdots & m_{i-1,n} \\ m_{i+1,1} & \cdots & m_{i+1,j-1} & m_{i+1,j+1} & \cdots & m_{i+1,n} \\ \vdots & & \vdots & \vdots & & \vdots \\ m_{n,1} & \cdots & m_{n,j-1} & m_{n,j+1} & \cdots & m_{n,n} \end{vmatrix}$$

where M_ij is called the minor of entry m_i,j in M. As an illustration, consider the following example:

$$\begin{aligned} \det\begin{bmatrix} 2 & -3 & 1 \\ 2 & 0 & -1 \\ 1 & 4 & 5 \end{bmatrix} &= 2\cdot\begin{bmatrix} 0 & -1 \\ 4 & 5 \end{bmatrix} - (-3)\cdot\begin{bmatrix} 2 & -1 \\ 1 & 5 \end{bmatrix} + 1\cdot\begin{bmatrix} 2 & 0 \\ 1 & 4 \end{bmatrix} \\ &= 2[0-(-4)] + 3[10-(-1)] + 1[8-0] \\ &= 2(0+4) + 3(10+1) + 1(8-0) \\ &= 2(4) + 3(11) + 1(8) \\ &= 8 + 33 + 8 \\ &= 49 \end{aligned}$$

(b) Write a recursive function with the following prototype:

```
double determinant(double **M, int n, int &count);
```

to calculate the determinant of the given nxn matrix M (n≥2). Upon return, the function sets variable count as the number of times that the recursive function is called(including the first call of the function) to complete the task of determinant calculation.Note that the function can calculate the determinant directly (using the well-known formula for a 2x2 matrix) when n=2 and hence count=1 in this case. For n≥3, the function should calculate the determinant using recursive function calls.

以下為程式內容

僅須實作並上傳 `//TEMPLATE BEGIN` 和 `//TEMPLATE END` 括起來的部分

`double** minor_matrix(double **M, int n, int i, int j);` is the function of Problem 5(a).

```
//PREPEND BEGIN
#include<iostream>

using namespace std;

double** minor_matrix(double **M, int n, int i, int j);
double determinant(double **M, int n, int &count);

int main()
{
    int N;
    cin>>N;
    double** M;
    M = new double*[N];
    for (int i = 0; i < N; ++i){
        M[i] = new double[N];
        for (int j = 0; j < N; ++j)
            cin >> M[i][j];
    }
    int count = 0;
    cout << determinant(M, N, count) << endl;
    cout << count << endl;
    for (int i = 0; i < N; ++i)
        delete[] M[i];
    delete[] M;
    return 0;
}
//PREPEND END

//TEMPLATE BEGIN
double** minor_matrix(double **M, int n, int i, int j){
    // TODO
}
double determinant(double **M, int n, int &count){
    // TODO
}
//TEMPLATE END
```

Input

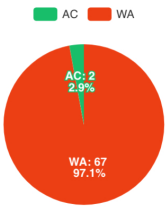
第一行為 n，代表矩陣的行數

Problems
Announcements
Submissions
Rankings
View Contest

Information

ID	5b
Time Limit	1000MS
Memory Limit	32MB
Created By	hungguo
Level	Low
Score	14
Tags	Show

Statistic Details



第一行輸入，代表矩陣的行數

然後以下為 n*n 的矩陣

Output

輸出矩陣的行列式和函式determinant的呼叫次數。

Sample Input 1

```
2
1 -1
-1 1
```

Sample Output 1

```
0
1
```

Sample Input 2

```
3
1 2 3
4 5 6
7 8 9
```

Sample Output 2

```
0
4
```

Language: C++ Theme: Solarized Light

1

You have solved the problem

Contest has ended

Submit