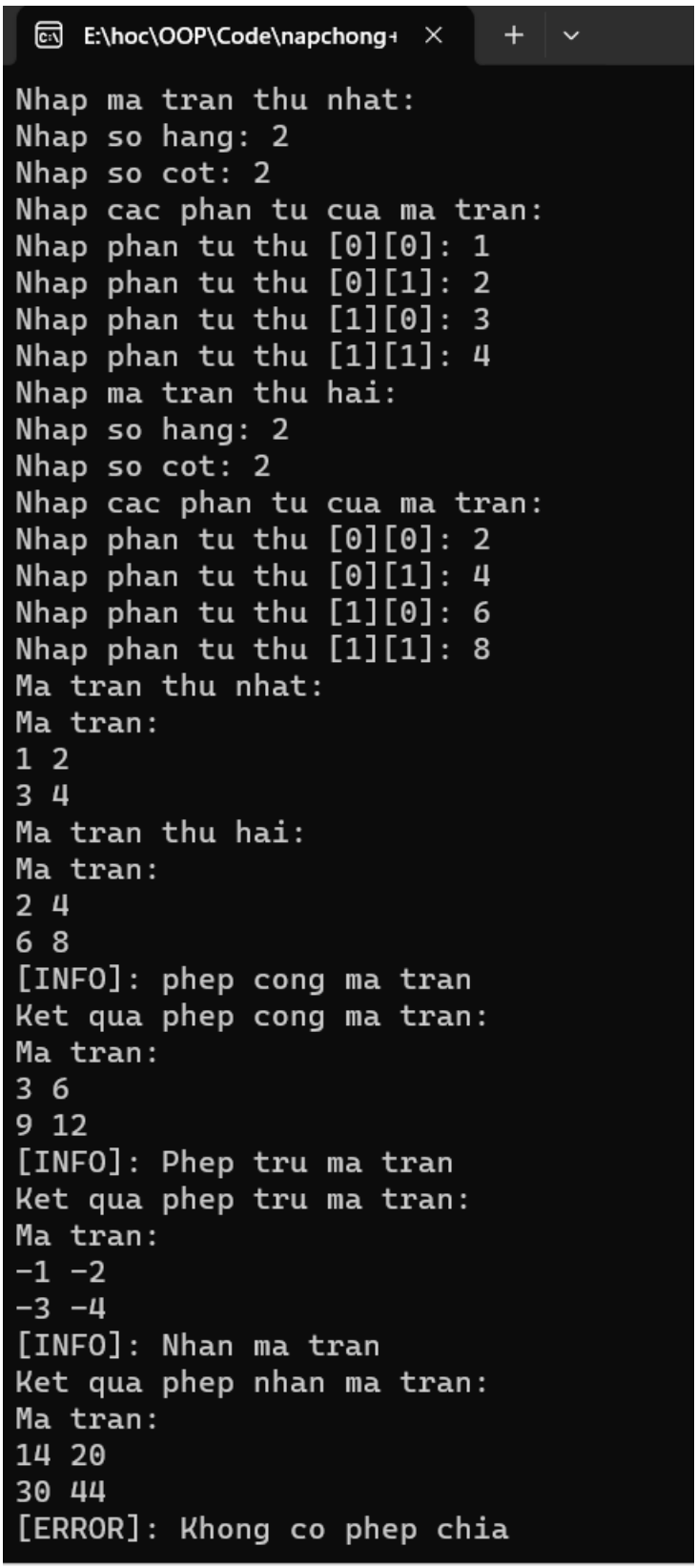
MÀN HÌNH CONSOLE:



```
Ma tran thu nhat:
Ma tran:
1 2
3 4
Ma tran thu hai:
Ma tran:
2 4
6 8
[INFO]: phep cong ma tran
Ket qua phep cong ma tran:
Ma tran:
3 6
9 12
[INFO]: Phep tru ma tran
Ket qua phep tru ma tran:
Ma tran:
-1 -2
-3 -4
[INFO]: Nhan ma tran
Ket qua phep nhan ma tran:
Ma tran:
14 20
30 44
[ERROR]: Khong co phep chia
[DEBUG]: Ban co the su dung ma tran nghich dao
Ket qua phep chia ma tran:

--------------------------------
Process exited after 9.761 seconds with return value 0
Press any key to continue . . .
```

CODE:

matrixlogging.cpp

```cpp
#include <iostream>
#include <vector>
#include<string>

using namespace std;

class Logger {
public:
    static void log(const string& message, int check) {
        if(check==0){
            cout <<"[INFO]: "<< message << endl;
        }else if(check==1){
            cout <<"[DEBUG]: "<< message << endl;
        }else if(check==2){
            cout <<"[ERROR]: "<< message << endl;
        }
    }
};

class Matrix {
private:
    int m;
    int n;
    double elements[100][100];

public:
    Matrix() : m(0), n(0) {}

    Matrix(const Matrix& a) : m(a.m), n(a.n) {
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                elements[i][j] = a.elements[i][j];
            }
        }
    }
```

matrixlogging.cpp

```cpp
    int m;
    int n;
    double elements[100][100];

public:
    Matrix() : m(0), n(0) {}

    Matrix(const Matrix& a) : m(a.m), n(a.n) {
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                elements[i][j] = a.elements[i][j];
            }
        }
    }

    void nhap() {
        cout << "Nhap so hang: ";
        cin >> m;
        cout << "Nhap so cot: ";
        cin >> n;

        cout << "Nhap cac phan tu cua ma tran:" << endl;
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                cout << "Nhap phan tu thu [" << i << "][" << j << "]: ";
                cin >> elements[i][j];
            }
        }
    }

    void xuat() const {
        cout << "Ma tran:" << endl;
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                cout << elements[i][j] << " ";
```

matrixlogging.cpp

```cpp
        }
    }

    void xuat() const {
        cout << "Ma tran:" << endl;
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                cout << elements[i][j] << " ";
            }
            cout << endl;
        }
    }

    Matrix operator+(const Matrix& a) const {
        Matrix result(*this);
        if (m == a.m && n == a.n) {
            for (int i = 0; i < m; i++) {
                for (int j = 0; j < n; j++) {
                    result.elements[i][j] += a.elements[i][j];
                }
            }
            Logger::log("phep cong ma tran",0);
            return result;
        }
        Logger::log("Khong the cong 2 ma tran khong cung kich thuoc",2);
        Logger::log("Vui long nhap lai ma tran",1);
        return *this;
    }

    Matrix operator-(const Matrix& a) const {
        Matrix result(*this);
        if (m == a.m && n == a.n) {
            for (int i = 0; i < m; i++) {
                for (int j = 0; j < n; j++) {
                    result.elements[i][j] -= a.elements[i][j];
```

matrixlogging.cpp

```cpp
        Matrix result(*this);
        if (m == a.m && n == a.n) {
            for (int i = 0; i < m; i++) {
                for (int j = 0; j < n; j++) {
                    result.elements[i][j] -= a.elements[i][j];
                }
            }
            Logger::log("Phep tru ma tran",0);
            return result;
        }
        Logger::log("Khong the tru 2 ma tran khong cung kich thuoc",2);
        Logger::log("Vui long nhap lai ma tran",1);
        return *this;
    }

    Matrix operator*(const Matrix& a) const {
        Matrix result;
        if (n == a.m) {
            result.m = m;
            result.n = a.n;
            for (int i = 0; i < m; i++) {
                for (int j = 0; j < a.n; j++) {
                    result.elements[i][j] = 0;
                    for (int k = 0; k < n; k++) {
                        result.elements[i][j] += elements[i][k] * a.elements[k][j];
                    }
                }
            }
            Logger::log("Nhan ma tran",0);
            return result;
        }
        Logger::log("Khong nhan duoc 2 ma tran khac kich thuoc",2);
        Logger::log("Vui long nhap lai ma tran",1);
        return result;
    }
```

matrixlogging.cpp

```
109         }
110         Logger::log("Khong nhan duoc 2 ma tran khac kich thuoc",2);
111         Logger::log("Vui long nhap lai ma tran",1);
112         return result;
113     }
114
115     Matrix operator/(const Matrix& a) const {
116         Logger::log("Khong co phep chia",2);
117         Logger::log("Ban co the su dung ma tran nghich dao",1);
118         return *this;
119     }
120 };
121
122 int main() {
123     Matrix mt1, mt2;
124     cout << "Nhap ma tran thu nhat:" << endl;
125     mt1.nhap();
126     cout << "Nhap ma tran thu hai:" << endl;
127     mt2.nhap();
128
129     cout << "Ma tran thu nhat:" << endl;
130     mt1.xuat();
131     cout << "Ma tran thu hai:" << endl;
132     mt2.xuat();
133
134     Matrix result;
135
136     result = mt1 + mt2;
137     cout << "Ket qua phep cong ma tran:" << endl;
138     result.xuat();
139
140     result = mt1 - mt2;
141     cout << "Ket qua phep tru ma tran:" << endl;
142     result.xuat();
143
```

matrixlogging.cpp

```
117             Logger::log("Ban co the su dung ma tran nghich dao",1);
118             return *this;
119         }
120 };
121
122 int main() {
123     Matrix mt1, mt2;
124     cout << "Nhap ma tran thu nhat:" << endl;
125     mt1.nhap();
126     cout << "Nhap ma tran thu hai:" << endl;
127     mt2.nhap();
128
129     cout << "Ma tran thu nhat:" << endl;
130     mt1.xuat();
131     cout << "Ma tran thu hai:" << endl;
132     mt2.xuat();
133
134     Matrix result;
135
136     result = mt1 + mt2;
137     cout << "Ket qua phep cong ma tran:" << endl;
138     result.xuat();
139
140     result = mt1 - mt2;
141     cout << "Ket qua phep tru ma tran:" << endl;
142     result.xuat();
143
144     result = mt1 * mt2;
145     cout << "Ket qua phep nhan ma tran:" << endl;
146     result.xuat();
147
148     result = mt1 / mt2;
149     cout << "Ket qua phep chia ma tran:" << endl;
150     return 0;
151 }
```