#include<iostream>

#include<iomanip>

#include<Eigen/Dense>

using namespace std;

using namespace Eigen;

const int MAX = 10;

typedef double m2c[MAX][MAX];

void inputMatrix(int &m,int &n,m2c A){

do{

cout << "Enter the row number of matrix A: "; cin >> m;

cout << "Enter the column number of matrix A: "; cin >> n;

} while(m > n);

cout << "Enter value of matrix A:" << endl;

for(int i=0; i<m; i++){

for(int j=0; j<n; j++){

cin >> A[i][j];

}

}

}

void outputMatrix(int m,int n,m2c A){

for(int i=0; i<m; i++){

for(int j=0; j<n; j++){

cout << setw(10) << A[i][j];

}

cout << endl;

}

}

void transposeMatrix(int m,int n,m2c A,m2c B)

{

for(int i=0; i<m; i++){

for(int j=0; j<n; j++){

B[j][i] = A[i][j];

}

}

}

// This code apply for m<n

void multiplyMatrix(int m,int n,int q, m2c a,m2c b,m2c c){

for(int i=0; i<m; i++){

for(int j=0; j<q; j++){

c[i][j] = 0;

for(int k=0; k<n;k++){

c[i][j] += a[i][k]\*b[k][j];

}

}

}

}

void singularValueDecomposition(int m,int n,m2c A){

m2c B,C,U,W,V,VT;

// A^T \* A

transposeMatrix(m,n,A,B);

multiplyMatrix(n,m,n,B,A,C);

cout <<endl << endl <<"Matrix S = A^T\*A ="<< endl;

outputMatrix(n,n,C);

//Tim tri rieng va vector rieng

MatrixXd A\_eigen(n,n);

for(int i=0; i<n; i++){

for(int j=0; j<n; j++){

A\_eigen(i,j) = C[i][j];

}

}

EigenSolver<MatrixXd> eigensolver(A\_eigen);

if (eigensolver.info() != Success) {

cout << "Can't calculate eigenValues and eigenVectors!'" << endl;

return;

}

double lamda[n];

for(int i = 0; i<n; i++){

lamda[i] = real(eigensolver.eigenvalues()(i));

if(fabs(lamda[i]) < 1e-10) lamda[i] = 0;

}

//Tinh ma tran V

for(int i=0; i<n; i++){

for(int j=0; j<n; j++){

V[i][j] = real(eigensolver.eigenvectors()(i,j));

}

}

for(int i=0; i<n-1; i++){

for(int j=i+1; j<n; j++){

if(lamda[i] < lamda[j]){

// xep lamda giam dan

double temp = lamda[i];

lamda[i] = lamda[j];

lamda[j] = temp;

// xep vector rieng da duoc chuan hoa tuong ung voi tri rieng

for(int k=0; k<n; k++){

double temp = V[k][i];

V[k][i] = V[k][j];

V[k][j] = temp;

}

}

}

}

//Tinh ma tran xich ma

for(int i=0; i<m; i++){

W[i][i] = sqrt(lamda[i]);

for(int j=0; j<n; j++){

if(i!=j) W[i][j] = 0;

}

}

//Tinh ma tran U

multiplyMatrix(m,n,n,A,V,U);

for(int i=0; i<m; i++){

for(int j=0; j<m; j++){

U[j][i] = 1.0/sqrt(lamda[i])\*U[j][i];

}

}

//Tinh ma tran V^T

transposeMatrix(n,n,V,VT);

cout<< endl<<"EigenValue of matrix S = ";

for(int i=0; i<n; i++) cout << lamda[i] << " ";

cout<< endl<<endl<< "Matrix U = " <<endl;

outputMatrix(m,m,U);

cout<< endl<<"Matrix xich ma ="<<endl;

outputMatrix(m,n,W);

cout<< endl<< "Matrix V^T =" <<endl;

outputMatrix(n,n,VT);

cout <<endl<<"Check SVD of Matrix A: " << endl;

m2c kq,k;

multiplyMatrix(m,m,n,U,W,kq);

multiplyMatrix(m,n,n,kq,VT,k);

cout<<"Matrix A after check (A = U\*W\*V^T)" <<endl;

outputMatrix(m,n,k);

}

int main(){

int m,n;

m2c A,B,C;

inputMatrix(m,n,A);

cout <<endl<<"Matrix A:" << endl;

outputMatrix(m,n,A);

singularValueDecomposition(m,n,A);

}

