đề giữa kì toán ứng dụng - 11fsfsdsf

Toán ứng dụng CNTT (Trường Đại học Bách Khoa - Đại học Đà Nẵng)

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

ĐỀ THI GIỮA HỌC KỲ VÀ BÀI LÀM

Tên học phần: Toán ứng dụng CNTT
Mã học phần: Hình thức thi: *Tự luận*
Đề số: **0001** Thời gian làm bài: 90 phút *(không kể thời gian chép/phát đề)*
Được sử dụng tài liệu khi làm bài.

Họ tên: Nguyễn Quang Thiên Anh **Lớp**: 22T_Nhat1 **MSSV**: 102220264

Sinh viên làm bài trực tiếp trên tệp này, lưu tệp với định dạng MSSV_HọTên.pdf và nộp bài thông qua MSTeam

Câu 1 (***2 điểm***): Cho số nguyên dương N= 11530. Viết chương trình bằng C/C++ có sử dụng hàm thực hiện:

- Tìm số M là ước số nguyên tố lớn nhất của N; tìm số lượng các số hoàn hảo nhỏ hơn M liệt kê và tính tổng của chúng.
- Tìm các số nguyên tố bé hơn M, liệt kê và tính tổng của chúng.

 # Trả lời: Dán code vào bên dưới:

```
#include<bits/stdc++.h>


using namespace std;

bool isPrime(int n) {
    if (n <= 1) return false;
    if (n == 2) return true;
    if (n % 2 == 0) return false;
    for (int i = 3; i <= sqrt(n); i += 2)
        if (n % i == 0) return false;
    return true;
}

bool isPerfect(int n) {
    int sum = 1;
    for (int i = 2; i <= sqrt(n); i++) {
        if (n % i == 0) {
            if (i == (n / i)) sum = sum + i;
            else sum = sum + (i + n / i);
        }
    }
    return (sum == n && n != 1);
}

int largestPrimeFactor(int n) {
    int maxPrime = -1;
    while (n % 2 == 0) {
        maxPrime = 2;
        n >>= 1;
```

```
    }
    for (int i = 3; i <= sqrt(n); i += 2) {
        while (n % i == 0) {
            maxPrime = i;
            n = n / i;
        }
    }
    if (n > 2) maxPrime = n;
    return maxPrime;
}

vector<int> perfectNumbers(int M) {
    vector<int> perfects;
    for (int n = 2; n < M; n++)
        if (isPerfect(n)) perfects.push_back(n);
    return perfects;
}

vector<int> primeNumbers(int M) {
    vector<int> primes;
    for (int n = 2; n < M; n++)
        if (isPrime(n)) primes.push_back(n);
    return primes;
}

int sumOfVector(vector<int> vec) {
    int sum = 0;
    for (int i = 0; i < vec.size(); i++)
        sum += vec[i];
    return sum;
}

int main() {
    int sl=0;
    int N = 11530;
    int M = largestPrimeFactor(N);
    vector<int> perfects = perfectNumbers(M);
    vector<int> primes = primeNumbers(M);

    cout << "Uoc so nguyen to lon nhat cua "<<N<<" la = " << M << endl;
    cout << "So hoan hao be hon "<<M<<" la: ";
    for (int i = 0; i < perfects.size(); i++)
        cout << perfects[i] << " ";
    cout << "\nTong cac so hoan hao be hon "<< M<< " la: "<< sumOfVector(perfects) <<
endl;

   cout << "\nCac so nguyen to be hon "<< M<< " la: ";
    for (int i = 0; i < primes.size(); i++)
    {
        cout << primes[i] << " ";
          sl++;
    }
```

```
     cout<<endl;
    cout<<"Tong so luong cac so nguyen to be hon "<<M<<" la: "<<sl;
        cout<<endl;
   cout << "\nTong Cac so nguyen to be hon "<< M<< " la: "<< sumOfVector(primes) <<
endl;

    return 0;
}
```

Trả lời: Dán kết quả thực thi vào bên dưới:

```
Uoc so nguyen to lon nhat cua 11530 la = 1153
So hoan hao be hon 1153 la: 6 28 496
Tong cac so hoan hao be hon 1153 la: 530

Cac so nguyen to be hon 1153 la: 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191
93 197 199 211 223 227 229 233 239 241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 337 347 349 353 359 367 373 379 383 389 397 401 409 419 421 431 433 439 443 449 4
7 461 463 467 479 487 491 499 503 509 521 523 541 547 557 563 569 571 577 587 593 599 601 607 613 617 619 631 641 643 647 653 659 661 673 677 683 691 701 709 719 727 733 739 74
 751 757 761 769 773 787 797 809 811 821 823 827 829 839 853 857 859 863 877 881 883 887 907 911 919 929 937 941 947 953 967 971 977 983 991 997 1009 1013 1019 1021 1031 1033 1
39 1049 1051 1061 1063 1069 1087 1091 1093 1097 1103 1109 1117 1123 1129 1151 Tong so luong cac so nguyen to be hon 1153 la: 190

Tong Cac so nguyen to be hon 1153 la: 99685
```

***Câu 2*: (2 điểm)** *Cho hệ phương trình đồng dư sau*

$$\begin{cases} x \equiv 1 \bmod 3 \\ x \equiv 3 \bmod 5 \\ x \equiv 4 \bmod 7 \\ x \equiv 5 \bmod 11 \end{cases}$$

- Viết chương trình C/C++ có sử dụng hàm giải hệ phương trình đồng dư trên.

```
#include<bits/stdc++.h>
using namespace std;

int gcd(int a, int b) {
    if(b == 0) return a;
    else return gcd(b , a % b);
}

int TrungHoa(int k , int m[] , int a[] , int X[] , int X_1[] , int &M) {
    M = 1;
    int ans = 0;
    for(int i = 0 ; i < k ; ++i)
        M = M * m[i];
    for(int i = 0 ; i < k ; ++i)
        X[i] = M / m[i];
    for(int i = 0 ; i < k ; ++i) {
        int j = 1;
        while((j * m[i] + 1) % X[i] != 0) {
            j++;
        }
```

```
        X_1[i] = (j * m[i] + 1) / X[i];
    }
    for(int i = 0 ; i < k ; ++i)
        ans = ans + (a[i] * X[i] * X_1[i]);
    return ans;
}
int main() {
    int k;
    int M;
    cout << "Nhap so luong phuong trinh can tinh : "; cin >> k;
    int m[k + 1] , a[k + 1] , X[k + 1] , X_1[k + 1];
    bool kt;
    for(int i = 0 ; i < k ; ++i) {
        cout << "a[" << i  + 1 << "] " << " = "; cin >> a[i];
        cout << "m[" << i + 1 << "] " << " = "; cin >> m[i];
        do {
            kt = true;
            for(int j = 0 ; j < i ; ++j)
                if(gcd(m[i] , m[j]) != 1) {
                    cout << "Nhap lai m[" << i + 1 << "] " << " = "; cin >> m[i];
                    kt = false;
                    break;
                }
        }
        while(!kt);
    }
    int i = 0;
    int ans = TrungHoa(k , m , a , X , X_1 , M);
    int x =  ans % M;
    cout << "Phuong trinh can tim la : " << x << " + " << "k" << " * " << M << '\n';

}
```

```
Nhap so luong phuong trinh can tinh : 4
a[1]  = 1
m[1]  = 3
a[2]  = 3
m[2]  = 5
a[3]  = 4
m[3]  = 7
a[4]  = 5
m[4]  = 11
Phuong trinh can tim la : 1138 + k * 1155
```

Câu 3 (***3 điểm***): Cho ma trận A. Viết chương trình bằng c/c++ có sử dụng hàm thực hiện phân rã ma trận A (có hàm kiểm tra điều kiện phân rã).

a) Phân rã **Cholesky** LDL^T ma trận A

Trả lời: Dán code vào bên dưới (bao gồm điều kiện của ma trận A nếu có):

```
#include <bits/stdc++.h>
using namespace std;

const int MAX = 100;

void Cholesky_Decomposition(int matrix[][MAX],
                                    int n)
{
    int lower[n][n];
    memset(lower, 0, sizeof(lower));

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

            if (j == i)
            {
                for (int k = 0; k < j; k++)
                    sum += pow(lower[j][k], 2);
                lower[j][j] = sqrt(matrix[j][j] -
                                        sum);
            } else {
                for (int k = 0; k < j; k++)
                    sum += (lower[i][k] * lower[j][k]);
                lower[i][j] = (matrix[i][j] - sum) /
                                    lower[j][j];
            }
        }
    }

    cout << setw(6) << " Lower Triangular"
        << setw(30) << "Transpose" << endl;
    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++)
            cout << setw(6) << lower[i][j] << "\t";
        cout << "\t";

        for (int j = 0; j < n; j++)
            cout << setw(6) << lower[j][i] << "\t";
        cout << endl;
    }
}

int main()
{
    int n = 3;
    int matrix[][MAX] = { { 1, 2, 3 },
                          { 2, 7, 3 },
                          { 3, 3, 3 } };
```

```
    Cholesky_Decomposition(matrix, n);
    return 0;
}
```

Trả lời: Dán kết quả thực thi vào bên dưới với $A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 7 & 3 \\ 3 & 3 & 3 \end{bmatrix}$ (sai số $\varepsilon = 10^{-5}$):

```
Lower Triangular                    Transpose
    1       0       0               1       2       3
    2       1       0               0       1      -3
    3      -3  -2147483648                  0       0  -2147483648
```

b) Phân rã **eigendecomposition** ma trận A

Trả lời: Dán code vào bên dưới (bao gồm điều kiện của ma trận A nếu có):

```
#include <iostream>
#include <iomanip>
#include <cmath>
#include <complex>
#include <Eigen/Dense>
#include <Eigen/Eigenvalues>
using namespace std;
using namespace Eigen;

double Det(MatrixXd A, int rows)
{
	double s = 1;
	double det;
	MatrixXd a(A.rows(), A.cols());

	if (rows == 1)
	{
		return A(0, 0);
	}
	else
	{
		det = 0;
		for (int c = 0; c < rows; c++)
		{
			int m = 0;
			int n = 0;
			for (int i = 0; i < rows; i++)
			{
				for (int j = 0; j < rows; j++)
				{
					a(i, j) = 0;
					if (i != 0 && j != c)
					{
						a(m, n) = A(i, j);
						if (n < (rows - 2))
							n++;
						else
						{
							n = 0;
							m++;
						}
					}
				}
			}
```

```
                    det = det + s * (A(0, c) * Det(a, rows - 1));
                    s = -1 * s;
             }
      }
      return det;
}



MatrixXd fixZero(MatrixXd A)
{
      for (int i = 0; i < A.rows(); i++)
      {
             for (int j = 0; j < A.cols(); j++)
             {
                    if (fabs(A(i, j)) < 1e-10)
                    {
                           A(i, j) = 0;
                    }
             }
      }
      return A;
}

MatrixXd inverse_Matrix(MatrixXd A)
{
      double r, a;
      MatrixXd inc(A.rows(), A.cols() * 2);
      for (int i = 0; i < inc.rows(); i++)
      {
             for (int j = 0; j < inc.cols(); j++)
             {
                    if (j < A.rows())
                           inc(i, j) = A(i, j);
                    else
                           inc(i, j) = (i == (j - A.rows())) ? 1 : 0;
             }
      }

      for (int i = 0; i < A.rows(); i++)
      {
             for (int j = 0; j < A.rows(); j++)
             {
                    if (i != j)
                    {
                           r = inc(j, i) / inc(i, i);
                           for (int k = 0; k < inc.cols(); k++)
                           {
                                  inc(j, k) -= r * inc(i, k);
                           }
                    }
             }
      }

      for (int i = 0; i < A.rows(); i++)
      {
             a = inc(i, i);
             for (int j = 0; j < inc.cols(); j++)
             {
                    inc(i, j) /= a;
             }
      }

      MatrixXd rs(A.rows(), A.cols());
      for (int i = 0; i < A.rows(); i++)
      {
             for (int j = 0; j < A.cols(); j++)
```

```
		{
			rs(i, j) = inc(i, j + A.cols());
		}
	}

	return rs;

}

MatrixXd multiply(MatrixXd A, MatrixXd B, int x, int n, int y)
{
	MatrixXd rs(x, y);
	rs.setZero();
	for (int i = 0; i < x; i++)
	{
		for (int j = 0; j < y; j++)
		{
			for (int k = 0; k < n; k++)
			{
				rs(i, j) += A(i, k) * B(k, j);
			}
		}
	}
	return rs;
}

int main()
{
	int row, col;

	cout << "Nhap so hang va so cot cho ma tran A: " << endl;
	cin >> row >> col;
	MatrixXd A(row, col);
	for (int i = 0; i < row; i++)
	{
		for (int j = 0; j < col; j++)
		{
			cout << "A[" << i + 1 << "][" << j + 1 << "] = ";
			cin >> A(i, j);
		}
	}
	cout << endl;


	cout << "Ma tran A luc ban dau: " << endl;
	cout << A << endl;

	EigenSolver<MatrixXd> solver(A);
	MatrixXd EigenVa = solver.eigenvalues().real().matrix();
	cout << "Cac eigenvalue la: " << endl << EigenVa << endl;

	cout << "Ma tran P = " << endl;
	MatrixXd EigenVe = fixZero(solver.eigenvectors().real());
	cout << EigenVe << endl;

	if (Det(EigenVe, EigenVe.rows() == 0))
	{
		cout << "Ma tran khong co nghich dao!" << endl;
		return 0;
	}

	cout << "Ma tran P^(-1) = " << endl;
	cout << fixZero(inverse_Matrix(EigenVe)) << endl;

	MatrixXd D(A.rows(), A.cols());
	D.setZero();
	for (int i = 0; i < D.rows(); i++)
```

```
            for (int j = 0; j < D.cols(); j++)
                  if (i == j)
                        D(i, j) = EigenVa(i);
      cout << "Ma tran D:" << endl << D << endl;
      cout << "P * D = " << endl;
      cout << multiply(EigenVe, D, EigenVe.rows(), EigenVe.cols(), D.cols()) << endl;
      cout << "A * P = " << endl;
      cout << multiply(A, EigenVe, A.rows(), A.cols(), EigenVe.cols()) << endl;
      cout << "P * D * P^(-1) = " << endl;
      cout << fixZero(multiply(multiply(EigenVe, D, EigenVe.rows(), EigenVe.cols(),
D.cols()),
            inverse_Matrix(EigenVe), EigenVe.rows(), EigenVe.cols(), EigenVe.cols())) <<
endl;

      return 0;
}
```

Trả lời: Dán kết quả thực thi vào bên dưới với $A=\begin{bmatrix}1 & 2 & 3\\ 6 & 7 & 3\\ 8 & 9 & 3\end{bmatrix}$ (sai số $\varepsilon = 10^{-5}$):

```
Ma tran A luc ban dau:
1 2 3
6 7 3
8 9 3
Cac eigenvalue la:
     13.3899
    -2.38987
-5.81242e-16
Ma tran P =
-0.278139 -0.720776  0.688247
-0.609657  0.254581 -0.688247
-0.742264  0.644724  0.229416
Ma tran P^(-1) =
-0.670265 -0.813028  -0.42829
-0.868609 -0.596738  0.815613
 0.272431 -0.953509  0.681078
Ma tran D:
     13.3899            0            0
           0     -2.38987            0
           0            0 -5.81242e-16
P * D =
    -3.72425      1.72256 -4.00038e-16
    -8.16323    -0.608416  4.00038e-16
    -9.93882     -1.54081 -1.33346e-16
A * P =
    -3.72425      1.72256 -2.44249e-15
    -8.16323    -0.608416 -1.55431e-15
    -9.93882     -1.54081 -1.55431e-15
P * D * P^(-1) =
1 2 3
6 7 3
8 9 3
```

Câu 4 (***3 điểm***): Cho ma trận A. Viết chương trình bằng c/c++ có sử dụng hàm thực hiện phân rã ma trận A bằng phương pháp SVD.

 # Trả lời: Dán code vào bên dưới (bao gồm điều kiện của ma trận A nếu có):

```cpp
#include<cmath>
#include<iostream>
#include<algorithm>
#include<vector>
#include<utility>
#include<string>
#include<math.h>
#include<numeric>
#include<iterator>
#include<Eigen/Eigenvalues>
#include<Eigen/Dense>
using namespace std;
using namespace Eigen;

void transpose(MatrixXd& V, MatrixXd& V1, int row, int col) {
    for (int i = 0; i < col; i++) {
        for (int j = 0; j < row; j++) {
            V1(i, j) = V(j, i);
```

```
        }
    }
}

void reverse_matrix(MatrixXd& matrix) {
    for (int j = 0; j < matrix.cols() / 2; ++j)
        for (int i = 0; i < matrix.rows(); ++i) {
            swap(matrix(i, j), matrix(i, matrix.cols() - 1 - j));
        }
}


double matrixMultiply(VectorXd v1, VectorXd v2, int n) {
    double ans = 0;
    if (v1.size() != v2.size()) return 1;
    else {
        for (int i = 0; i < n; ++i)
            ans = ans + (v1[i] * v2[i]);
        return ans;
    }
}
int main()
{
    int m, n;
    cout << "Number of rows: " << endl;
    cin >> m;
    cout << "Number of columns" << endl;
    cin >> n;
    MatrixXd A(m, n), At(n, m);
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            int x;
            cout << "A["<<i+1<<"]"<<"["<<j+1<<"]= ";
            cin >> x;
            A(i, j) = x;
        }
    }
    transpose(A, At, m, n);

    MatrixXd U(m, m), D(m, n), V(n, n), VT(n, n), ATA;
    U.setZero();
    VT.setZero();
    D.setZero();
    ATA = At * A;
    cout << ATA << '\n';
    SelfAdjointEigenSolver<MatrixXd> eigensolver(ATA);
    V = eigensolver.eigenvectors();

    reverse_matrix(V);

    MatrixXd hold1 = eigensolver.eigenvalues().matrix();
    vector<long double> a;

    for (int i = 0; i < hold1.size(); i++) {
        if (hold1(i, 0) > 10e-10) {
            a.push_back((hold1(i, 0)));
        }
    }
    sort(a.rbegin(), a.rend());
    int pos = a.size();
    for (int i = 0; i < a.size(); i++) {
        D(i, i) = sqrt(a[i]);
    }

    for (int i = 0; i < a.size(); i++) {
        MatrixXd Ui = (1.0 / D(i, i)) * A * V.col(i);
        for (int j = 0; j < m; j++) {
```

```
            U(j, i) = Ui(j, 0);
        }
    }



    while (pos < m) {
        MatrixXd e3(m, 1);
        for (int i = 0; i < m; i++) {
            e3(i, 0) = 0;
        }
        e3(pos, 0) = 1;
        MatrixXd Ui = e3;
        for (int i = 0; i < pos; i++) {
            MatrixXd ui = U.col(i);

            ui = (matrixMultiply(e3, ui, m) / matrixMultiply(ui, ui, m)) * ui;
            Ui = Ui - ui;
        }
        cout << Ui << endl;
        long double chuanhoa = 0;
        for (int i = 0; i < Ui.rows(); i++) {
            chuanhoa += Ui(i, 0) * Ui(i, 0);
        }
        for (int j = 0; j < m; j++) {
            U(j, pos) = Ui(j, 0) / sqrt(chuanhoa);
        }
        pos++;
    }
    cout << "Matrix U: " << endl << U << endl;
    cout << "Matrix D: " << endl << D << endl;
    cout << "Matrix V: " << endl << V << endl;

    transpose(V, VT, n, n);

    cout << "Matrix VT: " << endl << VT << endl;
    MatrixXd AA = U * D * VT;
    cout << "Matrix A = U * D * VT : " << endl << AA << endl;
    AA = AA - A;
    cout << "Test secure A - (U * D * VT): " << endl << AA << endl;
}
```

Trả lời: Dán kết quả thực thi vào bên dưới với $A = \begin{bmatrix} 1 & 2 & 3 \\ 6 & 7 & 3 \\ 8 & 9 & 3 \end{bmatrix}$ (sai số $\varepsilon = 10^{-5}$):

```
101 116  45
116 134  54
 45  54  27
 0.128205
-0.448718
 0.320513
Matrix U:
 0.184367  0.956414  0.226455
 0.605611  0.070924 -0.792594
 0.774109 -0.283272  0.566139
Matrix D:
16.0067       0       0
      0 2.40503       0
      0       0       0
Matrix V:
  0.625418  -0.367652   0.688247
  0.723132 -0.0582739  -0.688247
  0.293143   0.928136   0.229416
Matrix VT:
  0.625418   0.723132   0.293143
 -0.367652 -0.0582739   0.928136
  0.688247  -0.688247   0.229416
Matrix A = U * D * VT :
1 2 3
6 7 3
8 9 3
Test secure A - (U * D * VT):
 4.44089e-16            0 -4.44089e-16
           0  1.77636e-15 -8.88178e-16
           0  1.77636e-15 -8.88178e-16
```