|  |
| --- |
| 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: 21N12Hình thức thi: *Tự luận*

Đề số: **0005** Thời gian làm bài: 90 phút *(không kể thời gian chép/phát đề)*

Được sử dụng tài liệu khi làm bài.

**Họ tên:** Phùng Thị Ánh **Lớp**:21TCLC-DT2 **MSSV**: 102210151

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= 97700. 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 (giữa 1 và 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;  long long snt(int n) {  long long max=1;  for (int i = 2; i <= n; i++) {  while (n % i == 0) {  n /= i;  max = i;  }  }  cout << "M = " << max << "\n";  return max;  }  bool prime(long long n){  if (n==1) return false;  for (int i=2; i\*i<=n; i++)  if (n%i==0) return false;  return true;  }  long long sum(long long n){  if (n==1) return 0;  long long res=0;  for (int i=1; i<n; i++)  if (n%i==0) {  res+=i;  }  return res;  }  bool kt(long long n){  return n==sum(n);  }  void shh(long long n) {  long long d=0, t=0;  cout << "So hoan hao (giua 1 va M)";  for (int i=1; i<n; i++) {  if (kt(i)) {  cout << i << "\t";  t += i;  d++;  }  }  cout << "\nSo luong cac so hoan hao: " << d << "\n";  cout << "Tong cac so hoan hao: "<< t <<"\n";  }  void sntbe(long long m) {  long long s=0;  cout<<"Cac so nguyen to be hon M: \n";  for (int i=2; i<m; i++)  if (prime(i)) {  s+=i;  cout<<i<<" ";  }  cout<<"\n";  cout<<"Tong cac so nguyen to be hon M: "<<s;  }  int main(){  long long n=97700,m,s=0;    m = snt(n);  shh(m);    sntbe(m);    return 0;  }  **# Trả lời:** Dán kết quả thực thi vào bên dưới: |

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



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

|  |
| --- |
| **# Trả lời:** Dán code vào bên dưới:  // Chinese  #include <bits/stdc++.h>  using namespace std;  long long power1(long long x,long long n){  long long res=1;  do {  if (n%2==1) res=res\*x;  x=x\*x;  n/=2;  } while (n);  return res;  }  long long power2(long long x,long long n,long long mod){  long long res=1;  do {  if (n%2==1) res=(res\*x)%mod;  x=(x\*x)%mod;  n/=2;  } while (n);  return res%mod;  }  long long gcd(long long a,long long b){  if (b==0) return a;  return gcd(b,a%b);  }  long long Euler(long long n){  long long res=1;  int i=2;  do {  while (n%i!=0) i++;  int d=0;  while (n%i==0) {  d++;  n/=i;  }  res=res \* power1(i,d-1) \* (i-1);  } while (n!=1);  return res;  }  long long Modulo\_Inverse(long long a,long long m){  return power2(a,Euler(m)-1,m);  }  int main(){    long long n=5;  long long a[]={0,2,3,4,5,6};  long long m[]={0,5,7,9,11,13};  long long temp=m[1],M=m[1];  for (int i=2; i<=n; i++) {  temp=gcd(temp,m[i]);  M=M\*m[i];  if (temp!=1) {  cout<<"Vo nghiem";  break;  }  }  long long res=0;  for (int i=1; i<=n; i++) res = res + a[i] \* (M/m[i]) \* Modulo\_Inverse(M/m[i],m[i]);  cout<<"X = "<<res%M;  cout<<" \* k ( Voi k>0) ";  }  **# Trả lời:** Dán kết quả thực thi vào bên dưới: |

***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.

1. Phân rã LDLT 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;  bool Check\_Symmetric(float a[][MAX], int n){  for (int i=0; i<n-1; i++)  for (int j=i+1; j<n; j++)  if (a[i][j]!=a[j][i])  return false;  return true;  }  void Cholesky\_Decomposition(float matrix[][MAX], int n){  float lower[MAX][MAX];float lowerT[MAX][MAX];float mtLD[MAX][MAX];float mtkq[MAX][MAX];  float D[MAX][MAX];  memset(lower, 0, sizeof(lower));  memset(D, 0, sizeof(D));    for (int i = 0; i < n; i++) {  lower[i][i] = 1;  for (int j = 0; j <= i; j++){  float sum1 = 0, sum2 = 0;  for (int k = 0; k < j; k++){  sum1 += lower[j][k] \* lower[j][k] \* D[k][k];  sum2 += lower[i][k] \* lower[j][k] \* D[k][k];  }  D[j][j] = matrix[j][j] - sum1;  lower[i][j] = (matrix[i][j] - sum2) / D[j][j];  }  }  cout << setw(6) << " Lower Triangular" << setw(55) << "Diagonal matrix" << setw(55) << "Transpose matrix" << endl;  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++){  cout << setw(8) << setprecision(6) << lower[i][j] << "\t";  lowerT[i][j] = lower[j][i];  }  cout << "\t";  for(int j = 0; j < n; j++)  cout << setw(8) << setprecision(6) << D[i][j] << "\t";  cout << "\t";  for (int j = 0; j < n; j++)  cout << setw(8) << setprecision(6) << lower[j][i] << "\t";  cout << endl;  }  }    // Driver Code  int main(){  int n = 3;  float matrix[][MAX] = { {-10, 4, 7},  {4,-5,8},  {7,8,-9}};  if(Check\_Symmetric(matrix, n)){  cout << "The given matrix is symmetric" << endl;  Cholesky\_Decomposition(matrix, n);  }  else  cout << "The given matrix is not symmetric" << endl;  return 0;  }  **# Trả lời:** Dán kết quả thực thi vào bên dưới với  (sai số ): |

1. Phân rã **eigendecomposition** 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;  #define MAX 100  void swap(float &a, float &b)  {  float temp;  temp = a;  a = b;  b = temp;  }  void in\_matran(float A[][MAX], int n)  {  for (int i = 0;i < n; i++)  {  for (int j = 0; j < n; j++) cout << A[i][j] << " ";  cout << endl;    }  }  float det(float a[][MAX],int n)  {  int i, j, k, dem = 0, kt;  float b[MAX], h, kq = 1;  for(i = 0; i < n - 1; i++)  {  if(a[i][i] == 0)  {  kt = 0;  for(j = i + 1; j < n; j++)  {  if(a[i][j] != 0)  {  for(k = 0; k < n; k++) swap(a[k][i],a[k][j]);  dem++; kt++;  break;  }  }  if(kt == 0) return 0;  }  b[i] = a[i][i];  for(j = 0; j < n; j++) a[i][j] /= b[i];  for(j = i + 1; j < n; j++)  {  h = a[j][i];  for(k = 0;k < n; k++) a[j][k] = a[j][k] - h \* a[i][k];  }  }  b[n - 1] = a[n - 1][n - 1];  for(i = 0; i < n; i++) kq \*= b[i];  if(dem % 2 == 0) return kq;  return -kq;  }  float con(float a[][MAX], int n, int h, int c)  {  float b[MAX][MAX];  int x = -1, y;  for (int i = 0; i < n; i++)  {  if (i == h) continue;  x++; y = -1;  for (int j = 0; j < n; j++)  {  if (j == c) continue;  y++;  b[x][y] = a[i][j];  }  }  if ((h + c) % 2 == 0) return det(b, n-1);  return -det(b, n-1);  }  void nghichdao(float A[][MAX], float B[][MAX], int n)  {  float temp[MAX][MAX];  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++) temp[i][j] = A[i][j];  }  for (int i = 0;i < n; i++)  {  for (int j = 0; j < n; j++) B[i][j] = con(A,n,i,j);  }  for (int i = 0; i < n - 1; i++)  {  for (int j = i + 1; j < n; j++)  {  swap(B[i][j], B[j][i]);  }    }  float k = det(A,n);  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++) A[i][j] = temp[i][j];  }  for (int i = 0; i < n;i++)  {  for (int j = 0; j < n; j++) B[i][j] /= k ;  }  }  void nhan\_matran\_vuong(float A[][MAX], float B[][MAX], float C[][MAX], int n)  {  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++)  {  C[i][j] = 0;  for (int k = 0; k < n; k++)  {  C[i][j] += A[i][k] \* B[k][j];  }  }  }  }  void giaiptb3(float C[], float X[])  {  float delta, k;  delta = pow(C[1], 2) -3 \* C[0] \* C[2];  k = (9 \* C[0] \* C[1] \* C[2] - 2 \* pow(C[1], 3) - 27 \* pow(C[0], 2) \* C[3]) / (2 \* sqrt(pow(fabs(delta), 3)));//(2\*sqrt(abs(pow(delta,3))))  if (delta > 0)  {  if (fabs(k) <= 1)  {  X[0] = (2 \* sqrt(delta) \* cos((acos(k)/3)) - C[1]) / (3 \* C[0]);  X[1] = (2 \* sqrt(delta) \* cos((acos(k) / 3 - (2 \* M\_PI / 3)))- C[1]) / (3 \* C[0]);  X[2] = (2 \* sqrt(delta) \* cos((acos(k) / 3 + (2 \* M\_PI / 3)))- C[1]) / (3 \* C[0]);  }  if (fabs(k) > 1)  {  X[0] = ((sqrt(delta) \* fabs(k)) / (3 \* C[0] \* k)) \* (pow((fabs(k) + sqrt(pow(k, 2) - 1)), 1.0 / 3) + pow((fabs(k) - sqrt(pow(k, 2) - 1)), 1.0 / 3)) - (C[1] / (3 \* C[0]));  }  }  else if (delta == 0)  {  X[0] = (-C[1] - pow(-(pow(C[1], 3) - 27 \* C[0] \* C[0] \* C[3]), 1.0 / 3)) / (3 \* C[0]);  }  else X[0] = (sqrt(fabs(delta)) / (3 \* C[0])) \* (pow((k + sqrt(k \* k + 1)), 1.0 / 3) - pow(-(k - sqrt(k \* k + 1)), 1.0 / 3)) - (C[1] / (3 \* C[0]));  }  void Eigen\_Decomposition(float A[][MAX], int n)  {  float M[MAX][MAX], M1[MAX][MAX], B[MAX][MAX], C[MAX][MAX], mtPd[MAX][MAX], mtPd\_1[MAX][MAX];  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++)  {  if (j == i) C[i][j] = 1;  else C[i][j] = 0;  }  }  for (int k = n-2; k >= 0; k--)  {  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++)  {  if (i != k)  {  if (i == j)  {  M[i][j] = 1;  M1[i][j] = 1;  }  else {  M[i][j] = 0;  M1[i][j] = 0;  }  }  else  {  M1[i][j] = A[k + 1][j];  if (j == k) M[i][j] = 1 / A[k+1][k];  else M[i][j] = -A[k+1][j] / A[k+1][k];  }  }  }    nhan\_matran\_vuong(M1, A, B, n);  nhan\_matran\_vuong(B, M, A, n);  nhan\_matran\_vuong(C, M, B, n);  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++) C[i][j] = B[i][j];  }  }  float heso[MAX] = {1, -A[0][0], -A[0][1], -A[0][2]};  cout << endl;  float X[MAX];  giaiptb3(heso,X);    float D[MAX][MAX];  cout << "X=" << X[2] << endl;  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++)  {  if (i == j) D[i][j] = X[i];  else D[i][j] = 0;  }  }  float T[MAX][MAX];  for (int i = 0; i < n;i++)  {  for (int j = 0; j < n; j++)  {  T[i][j] = pow(X[j], n - i - 1);  }  }  float P[MAX][MAX] , P1[MAX][MAX];  nhan\_matran\_vuong(B,T,P,n);    for (int j = 0; j < n; j++)  {  float p = 0;  for (int i = 0; i < n; i++) p+= pow(P[i][j], 2);  for (int i = 0; i < n; i++) P[i][j] = P[i][j] / sqrt(p);  }    for (int i = 0; i < n; i++)  for (int j = 0; j < n; j++)  cout << setw(8) << setprecision(5) << fixed << P[i][j] << "\t";  cout << endl;    nghichdao(P, P1, n);  cout << "Eigen\_Decomposition: " << endl;  cout << setw(10) << "P"  << setw(55) << "D" << setw(55) << "P1" << endl;  for (int i = 0; i < n; i++) {  for (int j = 0; j < n; j++)  cout << setw(8) << setprecision(5) << fixed << P[i][j] << "\t";  cout << "\t";  for(int j = 0; j < n; j++)  cout << setw(8) << setprecision(5) << fixed << D[i][j] << "\t";  cout << "\t";  for (int j = 0; j < n; j++)  cout << setw(8) << setprecision(5) << fixed << P1[i][j] << "\t";  cout << endl;  }  }  int main()  {  int n = 3;  float evalue[n];  float A[][MAX] = { {-10, 4, 7},  {4,-5,8},  {7,8,-9}};    Eigen\_Decomposition(A,n);  return 0;  }  **# Trả lời:** Dán kết quả thực thi vào bên dưới với  (sai số ): |

***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):  #include <iostream>  #include <iomanip>  #include <Eigen/Dense>  #include <Eigen/Eigenvalues>  using namespace Eigen;  using namespace std;  float data[9] = {10,2,8,9,0,0,9,7,6};  int rows = 3, cols = 3;    struct mat  {  int sd, sc;  float coef[100][100];  };    struct mat createMatrix(int m, int n)  {  struct mat A;  int i, j;    A.sd = m;  A.sc = n;  for (i = 0; i < m; i++)  for (j = 0; j < n; j++)  A.coef[i][j] = 0;    return A;  }    void SVD()  {  MatrixXf matrix(rows, cols);  for (int i = 0; i < rows; i++)  {  for (int j = 0; j < cols; j++)  {  matrix(i, j) = data[i \* cols + j];  }  }    struct mat sig = createMatrix(rows, cols);  MatrixXf sigma(rows, cols);  MatrixXf matrixA = matrix.transpose() \* matrix; //A\_T\*A    SelfAdjointEigenSolver<MatrixXf> esA(matrixA);  int k = 0;  VectorXf temp = esA.eigenvalues(); //temp: gia tri rieng    for (int i = 0; i < temp.size(); i++)  {  if (fabs(temp[i]) < 10e-4)  temp[i] = 0;  temp[i] = sqrt(temp[i]);  sig.coef[temp.size() - i - 1][temp.size() - i - 1] = temp[i];  }    MatrixXf V = esA.eigenvectors().transpose();  struct mat Vt = createMatrix(cols, cols);  for (int i = 0; i < cols; i++)  {  for (int j = 0; j < cols; j++)  {  Vt.coef[cols - i - 1][j] = V(i, j);  }  }    MatrixXf matrixB = matrix \* matrix.transpose();  SelfAdjointEigenSolver<MatrixXf> esB(matrixB);  MatrixXf U = esB.eigenvectors();    for (int i = 0; i < U.cols() / 2; i++)  {  temp = U.col(U.cols() - i - 1);  U.col(U.cols() - i - 1) = U.col(i);  U.col(i) = temp.col(0);  }    for (int i = 0; i < cols; i++)  {  for (int j = 0; j < cols; j++)  {  V(i, j) = Vt.coef[i][j];  if (fabs(V(i, j)) < 10e-4)  V(i, j) = 0;  }  }    for (int i = 0; i < rows; i++)  {  for (int j = 0; j < cols; j++)  {  sigma(i, j) = sig.coef[i][j];  if (fabs(sigma(i, j)) < 10e-4)  sigma(i, j) = 0;  }  }    for (int i = 0; i < min(sigma.rows(), sigma.cols()); i++)  {  if (sigma(i, i) != 0)  {  temp = V.row(i);  temp = temp.transpose();  temp = matrix \* temp / sigma(i, i);  U.col(i) = temp.col(0);  }  }  for (int i = 0; i < rows; i++)  {  for (int j = 0; j < rows; j++)  {  if (fabs(U(i, j)) < 10e-4)  U(i, j) = 0;  }  }  cout << "\nSVD Decomposition A = U \* Sigma \* V^T :" << endl;  cout << setprecision(5) << fixed;  cout << "\nU\n\n" << U << endl;  cout << "\nSigma\n\n" << sigma << endl;  cout << "\nV^T\n\n" << V << endl;  cout << "Thu lai" << endl;  ///////////  MatrixXf mtKQ ;  if (V.cols() < U.cols()){  MatrixXf mtVnew(rows, cols);  for (int i = 0; i < V.cols(); i++)  {  for (int j = 0; j < U.cols(); j++)  {  if (j == U.cols()-1)  mtVnew(i, j) = 0;  else mtVnew(i, j) = V(i,j);  }  }  mtKQ = U\*sigma\*mtVnew;  }  else mtKQ = U\*sigma\*V;  for (int i = 0; i < cols; i++)  {  for (int j = 0; j < cols; j++)  {  if (fabs(mtKQ(i, j)) < 10e-4)  mtKQ(i, j) = 0;  cout << setw(7) << setprecision(4) << fixed << mtKQ(i,j) << "\t";  }  cout << endl;  }  }    int main()  {  SVD();    return 0;  }  **# Trả lời:** Dán kết quả thực thi vào bên dưới với  (sai số ): |