|  |
| --- |
| TRƯỜNG ĐẠI HỌC BÁCH KHOA  **KHOA Công Nghệ Thông Tin**  BỘ MÔN: Công Nghệ Phần Mềm |

**ĐỀ THI VÀ BÀI LÀM**

Tên học phần: Trí tuệ nhân tạo

Mã học phần: Hình thức thi: *Tự luận có giám sát*

Đề số: **01** Thời gian làm bài: 60 phút *(không kể thời gian chép/phát đề)*

Được sử dụng tài liệu khi làm bài.

**Họ tên:…….**....Lý Thanh Hải…………………

**Lớp**:………….19TCLC\_DT2…………………

**MSSV**:……….102190061……………...

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*** (*3 điểm*): Viết chương trình nhập N = 20000, thực hiện các công việc sau:

1. *(1 điểm)* Hãy viết hàm tính số các ước của N

|  |
| --- |
| **# Trả lời:** Dán code vào bên dưới:  // số các ước của N  long long soCacUocSo(long long N)  {      long long count = 0;      for(int i = 1 ; i <= N; i++)      {          if(N % i == 0)          {              count++;          }      }      return count;  }  int main()  {      long long N;      cout << "nhap N: ";      cin >> N;      cout << "so cac uoc so cua N la = " << soCacUocSo(N) << endl;  }  **# Trả lời:** Dán kết quả thực thi vào bên dưới: |

1. *(1 điểm)* Hãy viết hàm tính tổng các ước của N

|  |
| --- |
| **# Trả lời:** Dán code vào bên dưới  // hàm tính tổng các ước của N  long long sumUoc(long long N)  {      long long sum = 0;      for(int i = 1 ; i <= N ; i++)      {          if(N % i == 0)          {              sum += i;          }      }      return sum;  }  int main()  {      long long N;      cout << "nhap N: ";      cin >> N;      cout << "tong uoc N la = " << sumUoc(N) << endl;  }  **# Trả lời:** Dán kết quả thực thi vào bên dưới: |

1. *(1 điểm)* Hãy viết hàm tính tích các ước của N

|  |
| --- |
| **# Trả lời:** Dán code vào bên dưới  // hàm tính tích các ước của N  long long multipleUoc(long long N)  {      long long multiple = 1;      for(int i = 1 ; i <= N/2; i++)      {          if(N % i == 0)          {              multiple \*= i;              multiple %= 1000000009;          }      }      return (multiple % 1000000009) \* N ;  }  int main()  {      long long N;      cout << "nhap N: ";      cin >> N;      cout << "tich cac uoc N la = " << multipleUoc(N) << endl;  }  **# Trả lời:** Dán kết quả thực thi vào bên dưới: |

***Câu 2*** (3 *điểm*): Phân rã ma trận A

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1. *(1 điểm)* Trình bày điều kiện để ma trận A có thể phân rã bằng SVD  |  | | --- | | **# Trả lời:** điều kiện của ma trận A  - ma trận A có thể là ma trận bất kỳ, đều có thể phân rã như ma trận không vuông , cholesky | |  |  1. (2 điểm) Mô tả thuật toán hoặc hàm thực thi phân rã SVD  |  | | --- | | **# Trả lời:** viết mô tả thuật toán hoặc dán code vào bên dưới  #include "../Eigen/Eigenvalues"  #include <iostream>  #include<complex>  using namespace std;  int main () {    int n =0,m=0;    cout << "nhap hang cua ma tran n = ";    cin >> n;    cout << "nhap cot cua ma tran m = ";    cin >> m;        Eigen::MatrixXd A(n,m);    Eigen::MatrixXd B(m,n);    Eigen::MatrixXd C(n,n);    Eigen::MatrixXd D(m,m);    // nhập ma trận A    for (int i = 0; i < n; i++)      {        for (int j = 0; j < m; j++)        {          cout << "nhap a[" << i << "][" << j << "]: ";          cin >> A(i,j);        }      }      cout << "matrix A:" << endl << A << endl;     // ma trận chuyển vị    for(int i = 0; i < n; i++)    {      for(int j = 0; j < m; j++)      {        B(j,i) = A(i,j);      }    }    for(int i = 0; i < n; i++)    {      for(int j = 0; j < n; j++)      {       C(i,j)=0;       for(int l = 0; l < m; l++)       {         C(i,j) += B(l,j) \* A(i,l);       }       cout << C(i,j)<<endl;      }    }    for(int i = 0; i < m; i++)    {      for(int j = 0; j < m; j++)      {        D(i,j)=0;        for(int l = 0; l < n; l++)        {            D(i,j) += B(i,l) \* A(l,j);        }      }    }    Eigen::EigenSolver < Eigen::MatrixXd> s (C);    Eigen::EigenSolver < Eigen::MatrixXd> t (D);    cout<<"B:"<<endl;    cout<<B<<endl;      cout<< s.eigenvalues() <<endl;      cout << "U:"<< endl;    cout << (s.eigenvectors()) << endl;    cout << "V:" << endl;    cout<< (t.eigenvectors());  } | |

***Câu 3***(4 *điểm*): Cho hàm f(*x*) = , hãy viết chương trình tìm giá trị nhỏ nhất nhỏ nhất của f(x) sử dụng thuật toán Gradient Descent Method

|  |
| --- |
| **# Trả lời**: Dán code vào bên dưới  #include<iostream>  #include<math.h>  using namespace std;  const int e = 2.71828;  // hàm f(x)  double cost(int x)  {      return pow(pow(e, x) - 2/pow(e, x), 2);  }  // đạo hàm f'(x)  double grad(int x)  {      return 2 \* pow(e, 2\*x) - 8 \* pow(e, -2 \* x);  }  double grad(double alpha, int x0, double gra = 1 \* pow(10, -3), int loop = 100)  {      double arr[loop];      arr[0] = x0;      double min = arr[0];      for(int i = 0 ; i < loop; i++)      {          arr[i + 1] = arr[i] - alpha \* grad(arr[i]);          if(min > arr[i + 1])          {              min = arr[i + 1];          }          if(abs(grad(i + 1)) < gra)          {              break;          }      }      // for(int i = 0 ; i < 100 ; i++)      // {      //     cout << arr[i] << " ";      // }      return min;  }  int main()  {      cout << "gia tri nho nhat la: " << endl;      cout << grad(0.1, 10);  }  #  **Trả lời:** Dán kết quả thực thi vào bên dưới: |

Đà Nẵng, ngày 19 tháng 10 năm 2021

|  |  |
| --- | --- |
| **GIẢNG VIÊN BIÊN SOẠN ĐỀ THI** | **TRƯỞNG BỘ MÔN** |
|  | (đã duyệt) |