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

**ĐỀ THI VÀ BÀI LÀM**

Tên học phần: Toán ứng dụng

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

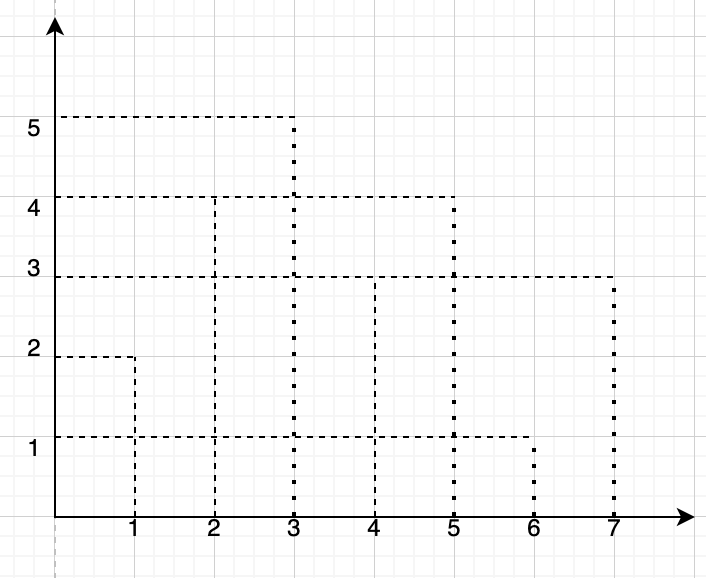
Đề số: **04** Thời gian làm bài: 60 phút *(không kể thời gian chép/phát đề)*

Được sử dụng tài liệu khi làm bài.

**Họ tên:** Đỗ Tân Trường Sanh **Lớp**: 19TCLC – DT2 **MSSV**: 102190087.

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*): Cho 7 điểm trong không gian Oxy với tọa độ như hình vẽ bên dưới



1. (1 điểm) Mô tả thuật toán xác định bao lồi từ

|  |
| --- |
| **# Trả lời:** viết mô tả thuật toán  B1 : Đầu tiên, ta xác định một điểm mà chắc chắn thuộc bao lồi. Gọi điểm này là điểm O  B2 : Chọn hệ trục tọa độ có gốc là điểm vừa chọn, đổi các tọa độ các điểm còn lại theo hệ trục tọa độ mới  B3: Tiếp theo, ta sắp xếp các điểm còn lại theo thứ tự tăng dần của góc tạo bởi trục hoành theo chiều dương và vector OI với I là một trong các điểm còn lại.  B4: Ta xét các điểm theo thứ tự ta vừa sắp xếp, với mỗi điểm ta sửa lại bao lồi H Gọi điểm đầu tiên được cho vào bao lồi H là H1, điểm cuối cùng là Hh (ban đầu h=0). |

1. (2 điểm) Viết chương trình xác định bao lỗi

|  |
| --- |
| **# Trả lời:** dán code vào bên dưới  #include<bits/stdc++.h>  using namespace std;  class Point  {  public:  int x, y;    Point() { }  Point(int a, int b)  {  x = a;  y = b;  }  };  bool my(Point a, Point b)  {  if(a.y == b.y)  return a.x < b.x;  return a.y < b.y;  }  Point v[10000];  vector<Point> ans;  int n;  double tinh(Point v1, Point v2)  {  return v1.x\*v2.y - v1.y\*v2.x;  }  bool sosanh(Point a, Point b)  {  Point v1(a.x - v[0].x, a.y - v[0].y);  Point v2(b.x - v[0].x, b.y - v[0].y);  return tinh(v1, v2) > 0;  }  void baoloi()  {  ans.push\_back(v[0]);  for(int i = 1; i < n; i++)  {    if(ans.size() < 2) ans.push\_back(v[i]);  else  {  Point now = ans[ans.size() - 1];  Point pre = ans[ans.size() - 2];  Point v1(v[i].x - pre.x, v[i].y - pre.y);  Point v2(v[i].x - now.x, v[i].y - now.y);  if(tinh(v1, v2) < 0) ans.pop\_back();  ans.push\_back(v[i]);  }  }  }  int main()  {  cin >> n;  for(int i = 0; i < n; i++)  cin >> v[i].x >> v[i].y;  sort(v, v + n, my);  sort(v + 1, v + n, sosanh);  baoloi();  for(auto i : ans)  {  cout << i.x << " " << i.y << 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   * Phân rã bằng SVD thực thi được với mọi ma trận |  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  import numpy as np  def SVD\_function(a):  lamda, V = np.linalg.eig(a.T.dot(a))  s = np.sqrt(lamda)  U = np.zeros((a.shape[0], a.shape[0]))  for i in range(len(U)):  v = np.array([np.array(V[:,i])]).T  U[:,i] = a.dot(v / s[i]).T  D = np.zeros\_like(a)  for i in range(a.shape[0]):  D[i][i] = s[i]  return U, D, V  def main():  a = np.array([[7.3, 2.7, 0.1], [2.7, 20, 3.5], [0.1, 3.5, 2]])  print("------Matrix A------\n", a, "\n")  n, m = a.shape  U, D, V = SVD\_function(a)  UDVT = U.dot(D).dot(V.T)  print("------Matrix U------\n", U, "\n")  print("------Matrix D------\n", D, "\n")  print("------Matrix UDVT------\n", UDVT, "\n")    if \_\_name\_\_ == '\_\_main\_\_':  main() | |

***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  import math  import numpy as np  import matplotlib.pyplot as plt  from decimal import \*  getcontext().prec = 7  #def cost(x):  #return x\*2 + 5\*np.sin(x)  #def grad(x):  #return 2\*x + 5\*np.cos(x)  # f(x)  # f(x)  def cost(x):  return float( (np.exp(x) - 2 // np.exp((x)))\*\*2)  def grad(x):  return float (2 \* np.exp((2\*x)) - 8 \* np.exp(-(2\*x)))  def myGD1(alpha, x0, gra = 1e-3, loop = 1000):  x = [x0]  for i in range(loop):  x\_new = x[-1] - alpha\*grad(x[-1])  if abs(grad(x\_new)) < gra:  break  x.append(x\_new)  return (x, i)  if \_\_name\_\_ == '\_\_main\_\_':  (x1, it1) = myGD1(0.1, 0.5)  print('Solution x1 = %f, cost = %f, obtained after %d iterations'%(x1[-1], cost(x1[-1]), it1))  #  **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) |