**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**



**MÔN LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

**BÀI TẬP THỰC HÀNH 3**

GVHD: Nguyễn Ngọc Quí

Sinh viên thực hiện: Đoàn Nguyễn Minh Thư - 24521725

🙡🙢 Tp. Hồ Chí Minh, 04/2025 🙠🙣

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*……., ngày……...tháng……năm 2025*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

**MỤC LỤC**

[Bài tập 1 6](#_Toc196342348)

[Bài tập 2 10](#_Toc196342349)

[Bài tập 3 14](#_Toc196342350)

[Bài tập 4 18](#_Toc196342351)

[Bài tập 5 21](#_Toc196342352)

[Bài tập 6 25](#_Toc196342353)

[Bài tập 7 28](#_Toc196342354)

[Bài tập 8 31](#_Toc196342355)

[TỔNG HỢP ĐƯỜNG LINK CODE CÁC BÀI 35](#_Toc196342356)

**DANH MỤC HÌNH ẢNH**

[Hình 1 - Class diagram của lớp TamGiac 6](file:///D:\Thư\Đời%20sống%20sinh%20viên\Năm%20nhất\OOP\Thực%20hành\lap3\24521725_BTTH3.docx#_Toc196342481)

[Hình 2 - Class diagram của lớp DaGiac 10](file:///D:\Thư\Đời%20sống%20sinh%20viên\Năm%20nhất\OOP\Thực%20hành\lap3\24521725_BTTH3.docx#_Toc196342482)

[Hình 3 - Class diagram của lớp cDSPhanSo 15](file:///D:\Thư\Đời%20sống%20sinh%20viên\Năm%20nhất\OOP\Thực%20hành\lap3\24521725_BTTH3.docx#_Toc196342483)

[Hình 4 - Class diagram của lớp cArray 18](file:///D:\Thư\Đời%20sống%20sinh%20viên\Năm%20nhất\OOP\Thực%20hành\lap3\24521725_BTTH3.docx#_Toc196342484)

[Hình 5 - Class diagram của lớp cDaThuc 22](file:///D:\Thư\Đời%20sống%20sinh%20viên\Năm%20nhất\OOP\Thực%20hành\lap3\24521725_BTTH3.docx#_Toc196342485)

[Hình 6 - Class diagram của lớp cCandidate và cListCandidate 26](file:///D:\Thư\Đời%20sống%20sinh%20viên\Năm%20nhất\OOP\Thực%20hành\lap3\24521725_BTTH3.docx#_Toc196342486)

[Hình 7 – Class diagram của lớp cNhanVienVP 28](file:///D:\Thư\Đời%20sống%20sinh%20viên\Năm%20nhất\OOP\Thực%20hành\lap3\24521725_BTTH3.docx#_Toc196342487)

[Hình 8 - Class diagram của lớp cNhanVienSX 31](file:///D:\Thư\Đời%20sống%20sinh%20viên\Năm%20nhất\OOP\Thực%20hành\lap3\24521725_BTTH3.docx#_Toc196342488)

**DANH MỤC BẢNG**

[Bảng 1.1 - Diem.h 6](#_Toc196342498)

[Bảng 1.2 - Diem.cpp 7](#_Toc196342499)

[Bảng 1.3 - TamGiac.h 8](#_Toc196342500)

[Bảng 1.4 - TamGiac.cpp 9](#_Toc196342501)

[Bảng 1.5 - main.cpp 10](#_Toc196342502)

[Bảng 2.1 - Diem.h 11](#_Toc196342503)

[Bảng 2.2 - Diem.cpp 12](#_Toc196342504)

[Bảng 2.3 - DaGiac.h 12](#_Toc196342505)

[Bảng 2.4 - DaGiac.cpp 14](#_Toc196342506)

[Bảng 2.5 - main.cpp 14](#_Toc196342507)

[Bảng 3.1 - PhanSo.h 15](#_Toc196342508)

[Bảng 3.2 - PhanSo.cpp 16](#_Toc196342509)

[Bảng 3.3 - cDSPhanSo.h 16](#_Toc196342510)

[Bảng 3.4 - cDSPhanSo.cpp 18](#_Toc196342511)

[Bảng 3.5 - main.cpp 18](#_Toc196342512)

[Bảng 4.1 - SoNguyen.h 19](#_Toc196342513)

[Bảng 4.2 - SoNguyen.cpp 19](#_Toc196342514)

[Bảng 4.3 - cArray.h 20](#_Toc196342515)

[Bảng 4.4 - cArray.cpp 21](#_Toc196342516)

[Bảng 4.5 - main.cpp 21](#_Toc196342517)

[Bảng 5.1 - DonThuc.h 22](#_Toc196342518)

[Bảng 5.2 - DonThuc.cpp 23](#_Toc196342519)

[Bảng 5.3 - cDaThuc.h 23](#_Toc196342520)

[Bảng 5.4 - cDaThuc.cpp 25](#_Toc196342521)

[Bảng 5.5 - main.cpp 25](#_Toc196342522)

[Bảng 6.1 - cCandidate.h 26](#_Toc196342523)

[Bảng 6.2 - cCandidate.cpp 27](#_Toc196342524)

[Bảng 6.3 - cListCandidate.h 27](#_Toc196342525)

[Bảng 6.4 - cListCandidate.cpp 28](#_Toc196342526)

[Bảng 6.5 - main.cpp 28](#_Toc196342527)

[Bảng 7.1 - cNhanVienVP.h 29](#_Toc196342528)

[Bảng 7.2 - cNhanVienVP.cpp 29](#_Toc196342529)

[Bảng 7.3 - cListNhanVienVP.h 30](#_Toc196342530)

[Bảng 7.4 - cListNhanVien.cpp 31](#_Toc196342531)

[Bảng 7.5 - main.cpp 31](#_Toc196342532)

[Bảng 8.1 – cNhanVienSX.h 32](#_Toc196342533)

[Bảng 8.2 - cNhanVienSX.cpp 33](#_Toc196342534)

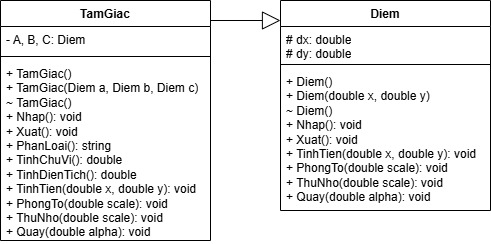
[Bảng 8.3 - cListNhanVienSX.h 33](#_Toc196342535)

[Bảng 8.4 - cListNhanVienSX.cpp 34](#_Toc196342536)

[Bảng 8.5 - main.cpp 34](#_Toc196342537)

**NỘI DUNG BÀI LÀM**

# Định nghĩa lớp cTamGiac để biểu diễn khái niệm tam giác trong mặt phẳng (gồm 3 đỉnh A, B, C) với các phương thức thiết lập, hủy bỏ (nếu có). Các hàm thành phần: thiết lập, nhập, xuất, kiểm tra loại tam giác, tính chu vi, tính diện tích, tịnh tiến, quay, phóng to, thu nhỏ.

Nội dung class diagram của chương trình được trình bày như hình 1.

Hình 1 - Class diagram của lớp TamGiac

Nội dung code của chương trình:

|  |
| --- |
| #**ifndef** DIEM\_H  #**define** DIEM\_H  #**include** <bits/stdc++.h>  **using** **namespace** std;  **class** Diem{  **protected**:  **double** dx;  **double** dy;  **public**:  Diem();  Diem(**double** x, **double** y);  ~Diem();  **void** **Nhap**();  **void** **Xuat**();  **void** **TinhTien**(**double** x, **double** y);  **void** **PhongTo**(**double** scale);  **void** **ThuNho**(**double** scale);  **void** **Quay**(**double** alpha);  **double** **getX**();  **double** **getY**();  };  #**endif** |

Bảng . - Diem.h

|  |
| --- |
| #**include** "Diem.h"  Diem::Diem(){  dx = 1;  dy = 1;  }  Diem::Diem(**double** x, **double** y){  dx = x;  dy = y;  }  Diem::~Diem(){  dx = 0;  dy = 0;  }  **void** Diem::Nhap(){  cout << "Nhap diem x: "; cin >> dx;  cout << "Nhap diem y: "; cin >> dy;  }  **void** Diem::Xuat(){  cout << "Toa do diem x: " << dx << " ";  cout << "Toa do diem y: " << dy << "\n";  }  **void** Diem::TinhTien(**double** x, **double** y){  dx += x;  dy += y;  }  **void** Diem::PhongTo(**double** scale){  dx \*= scale;  dy \*= scale;  }  **void** Diem::ThuNho(**double** scale){  dx /= scale;  dy /= scale;  }  **void** Diem::Quay(**double** alpha){  alpha = alpha\*M\_PI/180;  **double** xNew = dx\*cos(alpha) - dy\*sin(alpha);  **double** yNew = dx\*sin(alpha) + dy\*cos(alpha);  dx = xNew;  dy = yNew;  }  **double** Diem::getX(){  **return** **this**->dx;  }  **double** Diem::getY(){  **return** **this**->dy;  } |

Bảng . - Diem.cpp

|  |
| --- |
| #**include** "Diem.h"  **using** **namespace** std;  **class** TamGiac{  **private**:  Diem A, B, C;  **public**:  TamGiac();  TamGiac(Diem a, Diem b, Diem c);  ~TamGiac();  **void** **Nhap**();  **void** **Xuat**();  string **PhanLoai**();  **double** **TinhChuVi**();  **double** **TinhDienTich**();  **double** **TinhKhoangCach**(Diem a, Diem b);  **void** **TinhTien**(**double** x, **double** y);  **void** **PhongTo**(**double** scale);  **void** **ThuNho**(**double** scale);  **void** **Quay**(**double** alpha);  }; |

Bảng . - TamGiac.h

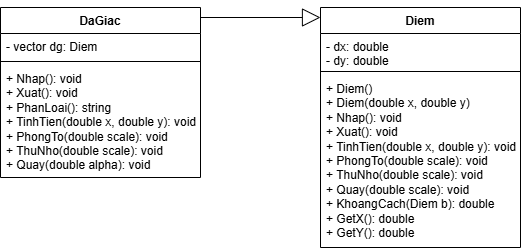
|  |
| --- |
| #**include** "TamGiac.h"  #**include** "Diem.h"  TamGiac::TamGiac(){  }  TamGiac::TamGiac(Diem a, Diem b, Diem c){  A = a;  B = b;  C = c;  }  TamGiac::~TamGiac(){  }  **void** TamGiac::Nhap(){  A.Nhap();  B.Nhap();  C.Nhap();  }    **void** TamGiac::Xuat(){  cout << "Toa do diem A la: "; A.Xuat();  cout << "Toa do diem B la: "; B.Xuat();  cout << "Toa do diem C la: "; C.Xuat();  }  **double** TamGiac::TinhKhoangCach(Diem a, Diem b){  **double** xA = a.getX(), xB = b.getX(), yA = a.getY(), yB = b.getY();  **double** ans = sqrt(pow(xB-xA, 2) + pow(yB-yA, 2));  **return** ans;  }  string TamGiac::PhanLoai(){  string ans = "Tam giac thuong";  **double** AB = TinhKhoangCach(A, B),  AC = TinhKhoangCach(A, C),  BC = TinhKhoangCach(B, C);  **double** AB2 = AB\*AB, AC2 = AC\*AC, BC2 = BC\*BC;  **if**(AB == BC && BC == AC) ans = "Tam giac deu";  **else** **if**(AB == BC || BC == AC || AB == AC) ans = "Tam giac can";  **else** **if**(AB2+AC2 == BC2 || AB2+BC2 == AC2 || BC2+AC2 == AB2) ans = "Tam giac vuong";  **return** ans;  }  **double** TamGiac::TinhChuVi(){  **double** AB = TinhKhoangCach(A, B),  AC = TinhKhoangCach(A, C),  BC = TinhKhoangCach(B, C);  **return** AB+AC+BC;  }  **double** TamGiac::TinhDienTich(){  **double** xA = A.getX(), yA = A.getY();  **double** xB = B.getX(), yB = B.getY();  **double** xC = C.getX(), yC = C.getY();  **return** 0.5\*fabs(xA\*(yB-yC) + xB\*(yC-yA) + xC\*(yA-yB));  }  **void** TamGiac::TinhTien(**double** x, **double** y){  A.TinhTien(x, y);  B.TinhTien(x, y);  C.TinhTien(x, y);  cout << "Tam giac sau khi tinh tien la: " << "\n";  Xuat();  }  **void** TamGiac::PhongTo(**double** scale){  A.PhongTo(scale);  B.PhongTo(scale);  C.PhongTo(scale);  cout << "Tam giac sau khi phong to la: " << "\n";  **this**->Xuat();  }  **void** TamGiac::ThuNho(**double** scale){  A.ThuNho(scale);  B.ThuNho(scale);  C.ThuNho(scale);  cout << "Tam giac sau khi thu nho la: " << "\n";  **this**->Xuat();  }  **void** TamGiac::Quay(**double** alpha){  A.Quay(alpha);  B.Quay(alpha);  C.Quay(alpha);  cout << "Tam giac sau khi quay la: " << "\n";  Xuat();  } |

Bảng . - TamGiac.cpp

|  |
| --- |
| #**include** <Diem.h>  #**include** <TamGiac.h>  **int** **main**(){  **double** tmp, x, y;  TamGiac tg;  tg.Nhap();  tg.Xuat();  cout << "Loai tam giac: " << tg.PhanLoai() << "\n";  cout << "Chu vi tam giac: " << tg.TinhChuVi() << "\n";  cout << "Dien tich tam giac: " << tg.TinhDienTich() << "\n";  cout << "Ban muon tinh tien x bao nhieu cm: "; cin >> x;  cout << "Ban muon tinh tien y bao nhieu cm: "; cin >> y;  tg.TinhTien(x, y);  cout << "Ban muon quay tam giac bao nhieu do: "; cin >> tmp;  tg.Quay(tmp);  cout << "Ban muon phong to tam giac len bao nhieu: "; cin >> tmp;  tg.PhongTo(tmp);  cout << "Ban muon thu nho tam giac lai bao nhieu: "; cin >> tmp;  tg.ThuNho(tmp);  **return** 0;  } |

Bảng . - main.cpp

# Định nghĩa lớp cDaGiac để biểu diễn khái niệm đa giác trong mặt phẳng với các hàm thành phần tương tự như lớp TamGiac.

Nội dung class diagram của chương trình được trình bày như hình 2.

Hình 2 - Class diagram của lớp DaGiac

Nội dung code của chương trình:

|  |
| --- |
| #**ifndef** DIEM\_H  #**define** DIEM\_H  #**include** <bits/stdc++.h>  **using** **namespace** std;  **class** Diem  {  **private**:  **double** dx, dy;  **public**:  Diem();  Diem(**double** x, **double** y);  **void** **Nhap**();  **void** **Xuat**();  **void** **TinhTien**(**double** x, **double** y);  **void** **PhongTo**(**double** scale);  **void** **ThuNho**(**double** scale);  **void** **Quay**(**double** alpha);  **double** **KhoangCach**(Diem b);  **double** **GetX**();  **double** **GetY**();  };  #**endif** // DIEM\_H |

Bảng . - Diem.h

|  |
| --- |
| #**include** "Diem.h"  Diem::Diem(){  }  Diem::Diem(**double** x, **double** y){  dx = x;  dy = y;  }  **void** Diem::Nhap(){  cout << "Nhap vao diem x: "; cin >> dx;  cout << "Nhap vao diem y: "; cin >> dy;  }  **void** Diem::Xuat(){  cout << "Toa do diem x: " << dx << " ";  cout << "Toa do diem y: " << dy << "\n";  }  **void** Diem::TinhTien(**double** x, **double** y){  dx += x;  dy += y;  }  **void** Diem::PhongTo(**double** scale){  dx \*= scale;  dy \*= scale;  }  **void** Diem::ThuNho(**double** scale){  dx /= scale;  dy /= scale;  }  **void** Diem::Quay(**double** alpha){  alpha = alpha\*M\_PI/180;  **double** xNew = dx\*cos(alpha) - dy\*sin(alpha);  **double** yNew = dx\*sin(alpha) + dy\*cos(alpha);  dx = xNew;  dy = yNew;  }  **double** Diem::KhoangCach(Diem b){  **double** ans = sqrt(pow(dx-b.dx, 2) + pow(dy-b.dy, 2));  **return** ans;  }  **double** Diem::GetX(){  **return** dx;  }  **double** Diem::GetY(){  **return** dy;  } |

Bảng . - Diem.cpp

|  |
| --- |
| #**include** "Diem.h"  **class** DaGiac  {  **private**:  vector<Diem>dg;  **public**:  **void** **Nhap**();  **void** **Xuat**();  string **PhanLoai**();  **void** **TinhTien**(**double** x, **double** y);  **void** **PhongTo**(**double** scale);  **void** **ThuNho**(**double** scale);  **void** **Quay**(**double** alpha);  }; |

Bảng . - DaGiac.h

|  |
| --- |
| #**include** "DaGiac.h"  **void** DaGiac::Nhap(){  **int** n;  cout << "So canh ban muon nhap la: "; cin >> n;  **for**(**int** i = 0; i < n; ++i){  Diem tmp;  tmp.Nhap();  dg.push\_back(tmp);  }  }  **void** DaGiac::Xuat(){  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].Xuat();  }  }  string DaGiac::PhanLoai(){  **int** n = dg.size();  **if**(n < 3) **return** "Khong phai da giac";  vector<**double**> canh(n);  **for**(**int** i = 0; i < n; ++i){  canh[i] = dg[i].KhoangCach(dg[(i+1)%n]);  }  **int** daGiacDeu = 1;  **for**(**int** i = 0; i < n-1; ++i){  **if**(canh[i] != canh[i-1]){  daGiacDeu = 0;  **break**;  }  }  **if**(daGiacDeu) **return** "Da giac deu";  **for**(**int** i = 0; i < n; ++i){  **int** a = (i-1+n) % n, b = i, c = (i+1)%n;  **double** v1x = dg[b].GetX() - dg[a].GetX();  **double** v1y = dg[b].GetY() - dg[a].GetY();  **double** v2x = dg[c].GetX() - dg[b].GetX();  **double** v2y = dg[c].GetY() - dg[b].GetY();  **double** dot = v1x \* v2x + v1y \* v2y;  **if** (fabs(dot) < 1e-6) {  **return** "Da giac vuong";  }  }  **for** (**int** i = 0; i < n; ++i) {  **for** (**int** j = i + 1; j < n; ++j) {  **if** (fabs(canh[i] - canh[j]) < 1e-6) {  **return** "Da giac can";  }  }  }  **return** "Da giac thuong";  }  **void** DaGiac::TinhTien(**double** x, **double** y){  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].TinhTien(x, y);  }  cout << "Da giac sau khi tinh tien la: " << "\n";  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].Xuat();  }  }  **void** DaGiac::PhongTo(**double** scale){  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].PhongTo(scale);  }  cout << "Da giac sau khi phong to la: " << "\n";  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].Xuat();  }  }  **void** DaGiac::ThuNho(**double** scale){  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].ThuNho(scale);  }  cout << "Da giac sau khi thu nho la: " << "\n";  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].Xuat();  }  }  **void** DaGiac::Quay(**double** alpha){  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].Quay(alpha);  }  cout << "Da giac sau khi quay la: " << "\n";  **for**(**int** i = 0; i < dg.size(); ++i){  dg[i].Xuat();  }  } |

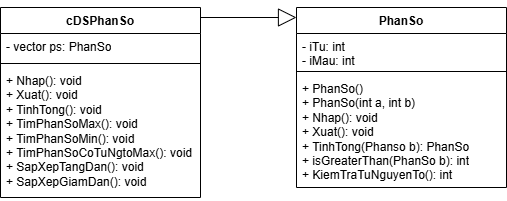
Bảng . - DaGiac.cpp

|  |
| --- |
| #**include** "Diem.h"  #**include** "DaGiac.h"  **int** **main**(){  DaGiac dg;  dg.Nhap();  dg.Xuat();  cout << "Da giac ban vua nhap thuoc loai: " << dg.PhanLoai() << "\n";  **double** x, y;  cout << "Ban muon tinh tien x len bao nhieu: "; cin >> x;  cout << "Ban muon tinh tien y len bao nhieu: "; cin >> y;  dg.TinhTien(x, y);  **double** scale;  cout << "Ban muon phong to da giac len may lan: "; cin >> scale;  dg.PhongTo(scale);  cout << "Ban muon thu nho da giac xuong may lan: "; cin >> scale;  dg.ThuNho(scale);  **double** alpha;  cout << "Ban muon quay da giac bao nhieu do: "; cin >> alpha;  dg.Quay(alpha);  **return** 0;  } |

Bảng . - main.cpp

# Định nghĩa lớp cDSPhanSo (danh sách phân số) để lưu trữ và xử lý các thao tác trên mảng phân số. Viết chương trình cho phép người dùng nhập vào danh sách các phân số: tính tổng, tìm phân số lớn nhất, tìm phân số nhỏ nhất, tìm phân số có tử số là nguyên tố lớn nhất, sắp xếp phân số tăng dần, sắp xếp phân số giảm dần.

Nội dung class diagram được trình bày như hình 3.

Nội dung code của chương trình:

Hình 3 - Class diagram của lớp cDSPhanSo

|  |
| --- |
| #**ifndef** PHANSO\_H  #**define** PHANSO\_H  #**include** <bits/stdc++.h>  **using** **namespace** std;  **class** PhanSo{  **private**:  **int** iTu;  **int** iMau;  **public**:  PhanSo();  PhanSo(**int** a, **int** b);  **void** **Nhap**();  **void** **Xuat**();  PhanSo **TinhTong**(PhanSo b);  **int** **isGreaterThan**(PhanSo b);  **int** **KiemTraTuNguyenTo**();  };  #**endif** // PHANSO\_H |

Bảng . - PhanSo.h

|  |
| --- |
| #**include** "PhanSo.h"  PhanSo::PhanSo(){  iTu = 0;  iMau = 1;  }  PhanSo::PhanSo(**int** a, **int** b){  iTu = a;  iMau = b;  }  **void** PhanSo::Nhap(){  cout << "Nhap tu so: "; cin >> iTu;  cout << "Nhap mau so: "; cin >> iMau;  }  **void** PhanSo::Xuat(){  cout << iTu << "/" << iMau << "\n";  }  PhanSo PhanSo::TinhTong(PhanSo b){  PhanSo tmp;  tmp.iTu = iTu\*b.iMau + b.iTu\*iMau;  tmp.iMau = iMau\*b.iMau;  **return** tmp;  }  **int** PhanSo::isGreaterThan(PhanSo b){  **int** tmp1, tmp2;  tmp1 = iTu\*b.iMau;  tmp2 = b.iTu\*iMau;  **if**(tmp1 > tmp2) **return** 1;  **return** 0;  }  **int** PhanSo::KiemTraTuNguyenTo(){  **int** n = iTu;  **if**(n < 2) **return** 0;  **for**(**int** i = 2; i < sqrt(n); ++i){  **if**(n%i == 0) **return** 0;  }  **return** 1;  } |

Bảng . - PhanSo.cpp

|  |
| --- |
| #**include** "PhanSo.h"  **class** cDSPhanSo{  **private**:  vector <PhanSo> ps;  **public**:  **void** **Nhap**();  **void** **Xuat**();  **void** **TinhTong**();  **void** **TimPhanSoMax**();  **void** **TimPhanSoMin**();  **void** **TimPhanSoCoTuNgtoMax**();  **void** **SapXepTangDan**();  **void** **SapXepGiamDan**();  }; |

Bảng . - cDSPhanSo.h

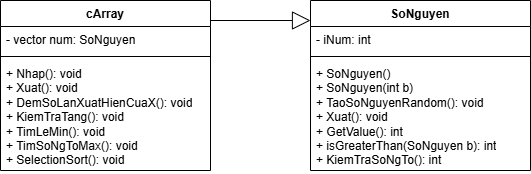
|  |
| --- |
| #**include** "cDSPhanSo.h"  **void** cDSPhanSo::Nhap(){  **int** n;  cout << "So phan so ban muon nhap la: "; cin >> n;  **for**(**int** i = 0; i < n; ++i){  PhanSo tmp;  tmp.Nhap();  ps.push\_back(tmp);  }  }  **void** cDSPhanSo::Xuat(){  **for**(**int** i = 0; i < ps.size(); ++i){  cout << "Phan so thu " << i+1 << " la: ";  ps[i].Xuat();  }  }  **void** cDSPhanSo::TinhTong(){  PhanSo tmp;  **for**(**int** i = 0; i < ps.size(); ++i){  tmp = tmp.TinhTong(ps[i]);  }  cout << "Ket qua sau khi tinh tong la: "; tmp.Xuat();  }  **void** cDSPhanSo::TimPhanSoMax(){  PhanSo mx;  mx = ps[0];  **for**(**int** i = 0; i < ps.size(); ++i){  **if**(mx.isGreaterThan(ps[i]) == 0) mx = ps[i];  }  cout << "Phan so lon nhat la: "; mx.Xuat();  }  **void** cDSPhanSo::TimPhanSoMin(){  PhanSo mn;  mn = ps[0];  **for**(**int** i = 0; i < ps.size(); ++i){  **if**(mn.isGreaterThan(ps[i]) == 1) mn = ps[i];  }  cout << "Phan so nho nhat la: "; mn.Xuat();  }  **void** cDSPhanSo::TimPhanSoCoTuNgtoMax(){  PhanSo mx;  **for**(**int** i = 0; i < ps.size(); ++i){  **if**(ps[i].KiemTraTuNguyenTo() == 1 && mx.isGreaterThan(ps[i]) == 0) mx = ps[i];  }  cout << "Phan so co tu so la nguyen to lon nhat la: "; mx.Xuat();  }  **void** cDSPhanSo::SapXepTangDan(){  **for**(**int** i = 0; i < ps.size(); ++i){  **int** mn = i;  **for**(**int** j = i+1; j < ps.size(); ++j){  **if**(ps[mn].isGreaterThan(ps[j]) == 1) mn = j;  }  swap(ps[mn], ps[i]);  }  cout << "Cac phan so sau khi xep tang dan la: " << "\n";  **for**(**int** i = 0; i < ps.size(); ++i){  ps[i].Xuat();  }  }  **void** cDSPhanSo::SapXepGiamDan(){  **for**(**int** i = 0; i < ps.size(); ++i){  **int** mx = i;  **for**(**int** j = i+1; j < ps.size(); ++j){  **if**(ps[mx].isGreaterThan(ps[j]) == 0) mx = j;  }  swap(ps[mx], ps[i]);  }  cout << "Cac phan so sau khi xep giam dan la: " << "\n";  **for**(**int** i = 0; i < ps.size(); ++i){  ps[i].Xuat();  }  } |

Bảng . - cDSPhanSo.cpp

|  |
| --- |
| #**include** "PhanSo.h"  #**include** "cDSPhanSo.h"  **int** **main**(){  cDSPhanSo ps;  ps.Nhap();  ps.Xuat();  ps.TinhTong();  ps.TimPhanSoMax();  ps.TimPhanSoMin();  ps.TimPhanSoCoTuNgtoMax();  ps.SapXepTangDan();  ps.SapXepGiamDan();  **return** 0;  } |

Bảng . - main.cpp

# Định nghĩa lớp Mảng một chiều (cArray) để lữu trữ danh sách các số nguyên và hỗ trợ xử lý các thao tác cơ bản trên mảng một chiều.

Nội dung class diagram được trình bày như hình 4.

Hình - Class diagram của lớp cArray

Nội dung code của chương trình:

|  |
| --- |
| #**ifndef** SONGUYEN\_H  #**define** SONGUYEN\_H  #**include** <bits/stdc++.h>  **using** **namespace** std;  **class** SoNguyen{  **private**:  **int** iNum;  **public**:  SoNguyen();  SoNguyen(**int** b);  **void** **TaoSoNguyenRandom**();  **void** **Xuat**();  **int** **GetValue**();  **int** **isGreaterThan**(SoNguyen b);  **int** **KiemTraSoNgto**();  };  #**endif** // SONGUYEN\_H |

Bảng . - SoNguyen.h

|  |
| --- |
| #**include** "SoNguyen.h"  SoNguyen::SoNguyen(){  iNum = 0;  }  SoNguyen::SoNguyen(**int** b){  iNum = b;  }  **void** SoNguyen::TaoSoNguyenRandom(){  iNum = std::rand() % 100 + 1;  }  **void** SoNguyen::Xuat(){  cout << iNum << " ";  }  **int** SoNguyen::GetValue(){  **return** iNum;  }  **int** SoNguyen::isGreaterThan(SoNguyen b){  **if**(iNum > b.iNum) **return** 1;  **return** 0;  }  **int** SoNguyen::KiemTraSoNgto(){  **if**(iNum < 2) **return** 0;  **for**(**int** i = 2; i < sqrt(iNum); ++i){  **if**(iNum % i == 0) **return** 0;  }  **return** 1;  } |

Bảng . - SoNguyen.cpp

|  |
| --- |
| #**include** "SoNguyen.h"  **class** cArray{  **private**:  vector <SoNguyen> num;  **public**:  **void** **Nhap**();  **void** **Xuat**();  **void** **DemSoLanXuatHienCuaX**();  **void** **KiemTraTang**();  **void** **TimLeMin**();  **void** **TimSoNgToMax**();  **void** **SelectionSort**();  }; |

Bảng . - cArray.h

|  |
| --- |
| #**include** "cArray.h"  **void** cArray::Nhap(){  **int** n;  cout << "Ban muon sinh bao nhieu phan tu: "; cin >> n;  **for**(**int** i = 0; i < n; ++i){  SoNguyen tmp;  tmp.TaoSoNguyenRandom();  num.push\_back(tmp);  }  }  **void** cArray::Xuat(){  **for**(**int** i = 0; i < num.size(); ++i){  num[i].Xuat();  }  cout << "\n";  }  **void** cArray::DemSoLanXuatHienCuaX(){  **int** cnt = 0, x;  cout << "Ban muon dem so lan xuat hien cua so nao: "; cin >> x;  **for**(**int** i = 0; i < num.size(); ++i){  **if**(num[i].GetValue() == x) ++cnt;  }  cout << x << " xuat hien " << cnt << " lan." << "\n";  }  **void** cArray::KiemTraTang(){  **int** check = 1;  **for**(**int** i = 0; i < num.size()-1; ++i){  **if**(num[i].isGreaterThan(num[i+1])){  check = 0;  **break**;  }  }  **if**(check) cout << "Mang tang dan" << "\n";  **else** cout << "Mang khong tang dan" << "\n";  }  **void** cArray::TimLeMin(){  **int** mn = INT\_MAX;  **for**(**int** i = 0; i < num.size(); ++i){  **int** Num = num[i].GetValue();  **if**(Num % 2 == 1 && Num < mn) mn = Num;  }  **if**(mn == INT\_MAX) cout << "Mang khong co phan tu le" << "\n";  **else** cout << "Phan le nho nhat trong mang la: " << mn << "\n";  }  **void** cArray::TimSoNgToMax(){  **int** mx = 0;  **for**(**int** i = 0; i < num.size(); ++i){  **if**(num[i].KiemTraSoNgto() == 1 && num[i].GetValue() > mx) mx = num[i].GetValue();  }  **if**(mx == 0) cout << "Mang khong co so nguyen to" << "\n";  **else** cout << "So nguyen to lon nhat trong mang la: " << mx << "\n";  }  **void** cArray::SelectionSort(){  **for**(**int** i = 0; i < num.size(); ++i){  **int** mn = i;  **for**(**int** j = i+1; j < num.size(); ++j){  **if**(num[mn].isGreaterThan(num[j])) mn = j;  }  swap(num[mn], num[i]);  }  cout << "Mang sau khi sap xep tang dan (Selection Sort) la: ";  **for**(**int** i = 0; i < num.size(); ++i){  num[i].Xuat();  }  } |

Bảng . - cArray.cpp

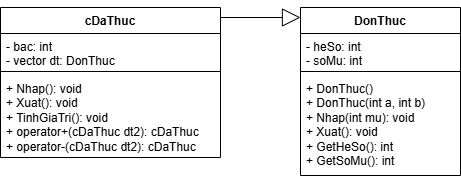
|  |
| --- |
| #**include** "SoNguyen.h"  #**include** "cArray.h"  **int** **main**(){  std::srand(std::time(nullptr));  cArray num;  num.Nhap();  num.Xuat();  num.DemSoLanXuatHienCuaX();  num.KiemTraTang();  num.TimLeMin();  num.TimSoNgToMax();  num.SelectionSort();  **return** 0;  } |

Bảng . - main.cpp

# Xét đa thức theo biến x (đa thực một biến) bậc n có dạng như sau:

Trong đó: n là bậc của đa thức và a1, a2, a3, ...., aj là các hệ số tương ứng với từng bậc của đa thức.

Định nghĩa lớp cDaThuc biểu diễn khái niệm đa thức với các thao tác.

Nội dung class diagram được trình bày như hình 5.

Hình - Class diagram của lớp cDaThuc

Nội dung code của chương trình:

|  |
| --- |
| #**ifndef** DONTHUC\_H  #**define** DONTHUC\_H  #**include** <bits/stdc++.h>  **using** **namespace** std;  **class** DonThuc  {  **private**:  **int** heSo;  **int** soMu;  **public**:  DonThuc();  DonThuc(**int** a, **int** b);  **void** **Nhap**(**int** mu);  **void** **Xuat**();  **int** **GetHeSo**();  **int** **GetSoMu**();  };  #**endif** // DONTHUC\_H |

Bảng . - DonThuc.h

|  |
| --- |
| #**include** "DonThuc.h"  DonThuc::DonThuc()  {  heSo = 1;  soMu = 0;  }  DonThuc::DonThuc(**int** a, **int** b){  heSo = a;  soMu = b;  }  **void** DonThuc::Nhap(**int** mu){  cout << "Nhap he so: "; cin >> heSo;  soMu = mu;  }  **void** DonThuc::Xuat(){  cout << heSo;  **if**(soMu != 0) cout << "x^" << soMu;  }  **int** DonThuc::GetHeSo(){  **return** heSo;  }  **int** DonThuc::GetSoMu(){  **return** soMu;  } |

Bảng . - DonThuc.cpp

|  |
| --- |
| #**include** "DonThuc.h"  **class** cDaThuc{  **private**:  **int** bac;  vector <DonThuc> dt;  **public**:  **void** **Nhap**();  **void** **Xuat**();  **void** **TinhGiaTri**();  cDaThuc **operator**+(cDaThuc dt2);  cDaThuc **operator**-(cDaThuc dt2);  }; |

Bảng . - cDaThuc.h

|  |
| --- |
| #**include** "cDaThuc.h"  **void** cDaThuc::Nhap(){  cout << "Ban muon nhap da thuc bac may: "; cin >> bac;  **for**(**int** i = bac; i >= 0; --i){  DonThuc x;  x.Nhap(i);  dt.push\_back(x);  }  }  **void** cDaThuc::Xuat(){  **for**(**int** i = 0; i < dt.size(); ++i){  **if**(dt[i].GetHeSo() == 0) **continue**;  **if**(i != 0){  **if**(dt[i].GetHeSo() > 0) cout << "+";  **else** cout << "-";  }  dt[i].Xuat();  }  cout << "\n";  }  **void** cDaThuc::TinhGiaTri(){  **long** **long** ans = 0;  **int** x;  cout << "Hay cho biet gia tri cua x: "; cin >> x;  **for**(**int** i = 0; i < dt.size(); ++i){  ans = ans + dt[i].GetHeSo()\*pow(x, dt[i].GetSoMu());  }  cout << "Gia tri cua da thuc la: " << ans << "\n";  }  cDaThuc cDaThuc::**operator**+(cDaThuc dt2){  **int** pos1 = 0, pos2 = 0;  cDaThuc Tong;  **while** (pos1 < dt.size() && pos2 < dt2.dt.size()) {  **int** mu1 = dt[pos1].GetSoMu();  **int** mu2 = dt2.dt[pos2].GetSoMu();  **if** (mu1 == mu2) {  **double** hsTong = dt[pos1].GetHeSo() + dt2.dt[pos2].GetHeSo();  **if** (hsTong != 0) Tong.dt.push\_back(DonThuc(hsTong, mu1));  pos1++; pos2++;  } **else** **if** (mu1 > mu2) {  Tong.dt.push\_back(dt[pos1]);  pos1++;  } **else** {  Tong.dt.push\_back(dt2.dt[pos2]);  pos2++;  }  }  **while** (pos1 < dt.size()) {  Tong.dt.push\_back(dt[pos1]);  pos1++;  }  **while** (pos2 < dt2.dt.size()) {  Tong.dt.push\_back(dt2.dt[pos2]);  pos2++;  }  **return** Tong;  }  cDaThuc cDaThuc::**operator**-(cDaThuc dt2){  **int** pos1 = 0, pos2 = 0;  cDaThuc Hieu;  **while** (pos1 < dt.size() && pos2 < dt2.dt.size()) {  **int** mu1 = dt[pos1].GetSoMu();  **int** mu2 = dt2.dt[pos2].GetSoMu();  **if** (mu1 == mu2) {  **double** hsHieu = dt[pos1].GetHeSo() - dt2.dt[pos2].GetHeSo();  **if** (hsHieu != 0) Hieu.dt.push\_back(DonThuc(hsHieu, mu1));  pos1++; pos2++;  } **else** **if** (mu1 > mu2) {  Hieu.dt.push\_back(dt[pos1]);  pos1++;  } **else** {  Hieu.dt.push\_back(dt2.dt[pos2]);  pos2++;  }  }  **while** (pos1 < dt.size()) {  Hieu.dt.push\_back(dt[pos1]);  pos1++;  }  **while** (pos2 < dt2.dt.size()) {  Hieu.dt.push\_back(dt2.dt[pos2]);  pos2++;  }  **return** Hieu;  } |

Bảng . - cDaThuc.cpp

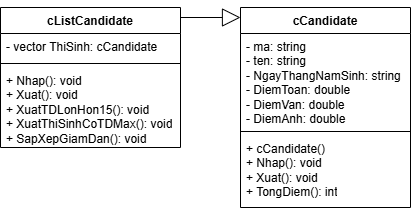
|  |
| --- |
| #**include** "DonThuc.h"  #**include** "cDaThuc.h"  **int** **main**(){  cDaThuc dt1, dt2;  dt1.Nhap();  dt1.Xuat();  dt1.TinhGiaTri();  cout << "Nhap da thuc thu 2: " << "\n";  dt2.Nhap();  cDaThuc Tong = dt1+dt2;  cout << "Da thuc sau khi cong la: ";  Tong.Xuat();  cDaThuc Hieu = dt1-dt2;  cout << "Da thuc sau khi tru la: ";  Hieu.Xuat();  **return** 0;  } |

Bảng . - main.cpp

# Định nghĩa lớp cCandidate (Thí sinh) gồm các thuộc tính: mã, tên, ngày tháng năm sinh, điểm thi Toán, Văn, Anh và các phương thức cần thiết.

Định nghĩa lớp cListCandidate (danh sách thí sinh) để lưu trữ và xử lý các thao tác trên danh sách thí sinh.

Nội dung class diagram được tình bày như hình 6.



Hình - Class diagram của lớp cCandidate và cListCandidate

Nội dung code của chương trình:

|  |
| --- |
| #**ifndef** CCANDIDATE\_H  #**define** CCANDIDATE\_H  #**include** <bits/stdc++.h>  **using** **namespace** std;  **class** cCandidate{  **private**:  string ma;  string ten;  string NgayThangNamSinh;  **double** DiemToan;  **double** DiemVan;  **double** DiemAnh;  **public**:  cCandidate();  **void** **Nhap**();  **void** **Xuat**();  **int** **TongDiem**();  };  #**endif** // CCANDIDATE\_H |

Bảng . - cCandidate.h

|  |
| --- |
| #**include** "cCandidate.h"  cCandidate::cCandidate(){  ma = "0000";  ten = "Khong ten";  NgayThangNamSinh = "Khong co";  DiemToan = DiemAnh = DiemVan = 0;  }  **void** cCandidate::Nhap(){  cin.ignore();  cout << "Ma thi sinh: "; getline(cin, ma);  cout << "Ho va ten thi sinh: "; getline(cin, ten);  cout << "Ngay thang nam sinh: "; getline(cin, NgayThangNamSinh);  cout << "Diem Toan: "; cin >> DiemToan;  cout << "Diem Anh: "; cin >> DiemAnh;  cout << "Diem Van: "; cin >> DiemVan;  }  **void** cCandidate::Xuat(){  cout << ma << " " << ten << " " << NgayThangNamSinh << " " << DiemToan << " " << DiemAnh << " " << DiemVan << "\n";  }  **int** cCandidate::TongDiem(){  **int** ans = DiemToan + DiemAnh + DiemVan;  **return** ans;  } |

Bảng . - cCandidate.cpp

|  |
| --- |
| #**include** "cCandidate.h"  **class** cListCandidate{  **private**:  vector <cCandidate> ThiSinh;  **public**:  **void** **Nhap**();  **void** **Xuat**();  **void** **XuatTDLonHon15**();  **void** **XuatThiSinhCoTDMax**();  **void** **SapXepGiamDan**();  }; |

Bảng . - cListCandidate.h

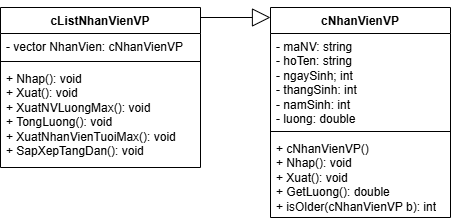
|  |
| --- |
| #**include** "cListCandidate.h"  **void** cListCandidate::Nhap(){  **int** n;  cout << "So luong thi sinh la: "; cin >> n;  **for**(**int** i = 0; i < n; ++i){  cCandidate x;  x.Nhap();  ThiSinh.push\_back(x);  }  }  **void** cListCandidate::Xuat(){  **for**(**int** i = 0; i < ThiSinh.size(); ++i){  ThiSinh[i].Xuat();  }  }  **void** cListCandidate::XuatTDLonHon15(){  **int** cnt = 0;  cout << "\n";  cout << "Danh sach thi sinh co tong diem lon hon 15 la: " << "\n";  **for**(**int** i = 0; i < ThiSinh.size(); ++i){  **if**(ThiSinh[i].TongDiem() > 15){  ++cnt;  ThiSinh[i].Xuat();  }  }  **if**(cnt == 0) cout << "Khong co" << "\n";  }  **void** cListCandidate::XuatThiSinhCoTDMax(){  **int** mx = ThiSinh[0].TongDiem();  **for**(**int** i = 1; i < ThiSinh.size(); ++i){  **if**(ThiSinh[i].TongDiem() > mx) mx = ThiSinh[i].TongDiem();  }  cout << "Thi sinh co tong diem cao nhat la: " << "\n";  **for**(**int** i = 0; i < ThiSinh.size(); ++i){  **if**(ThiSinh[i].TongDiem() == mx) ThiSinh[i].Xuat();  }  }  **void** cListCandidate::SapXepGiamDan(){  **for**(**int** i = 0; i < ThiSinh.size(); ++i){  **int** mx = i;  **for**(**int** j = i+1; j < ThiSinh.size(); ++j){  **if**(ThiSinh[j].TongDiem() > ThiSinh[mx].TongDiem()) mx = j;  }  swap(ThiSinh[mx], ThiSinh[i]);  }  cout << "Danh sach thi sinh sau khi sap xep giam dan theo tong diem la: " << "\n";  Xuat();  } |

Bảng . - cListCandidate.cpp

|  |
| --- |
| #**include** "cCandidate.h"  #**include** "cListCandidate.h"  **int** **main**(){  cListCandidate ThiSinh;  ThiSinh.Nhap();  ThiSinh.Xuat();  ThiSinh.XuatTDLonHon15();  ThiSinh.XuatThiSinhCoTDMax();  ThiSinh.SapXepGiamDan();  **return** 0;  } |

Bảng . - main.cpp

# Cài đặt lớp Nhân viên văn phòng (cNhanVienVP) cho phép lưu trữ các thông tin: mã nhân viên, họ tên, ngày sinh, lương và các thao tác cần thiết.

Nội dung class diagram được trình bày như hình 7.

Hình – Class diagram của lớp cNhanVienVP

Nội dung code của chương trình:

|  |
| --- |
| #**ifndef** CNHANVIENVP\_H  #**define** CNHANVIENVP\_H  #**include** <bits/stdc++.h>  **using** **namespace** std;  **class** cNhanVienVP  {  **private**:  string maNV, hoTen;  **int** ngaySinh, thangSinh, namSinh;  **double** luong;  **public**:  cNhanVienVP();  **void** d**Nhap**();  **void** **Xuat**();  **double** **GetLuong**();  **int** **isOlder**(cNhanVienVP b);  };  #**endif** // CNHANVIENVP\_H |

Bảng . - cNhanVienVP.h

|  |
| --- |
| #**include** "cNhanVienVP.h"  cNhanVienVP::cNhanVienVP(){  maNV = "Khong co";  hoTen = "Khong co";  ngaySinh = 0;  thangSinh = 0;  namSinh = 0;  luong = 0;  }  **void** cNhanVienVP::Nhap(){  cin.ignore();  cout << "Ma NV: "; getline(cin, maNV);  cout << "Ho va ten NV: "; getline(cin, hoTen);  cout << "Ngay sinh cua NV: "; cin >> ngaySinh;  cout << "Thang sinh cua NV: "; cin >> thangSinh;  cout << "Nam Sinh cua NV: "; cin >> namSinh;  cout << "Luong NV: "; cin >> luong;  }  **void** cNhanVienVP::Xuat(){  cout << maNV << " " << hoTen << " " << ngaySinh << "/" << thangSinh << "/" << namSinh << " " << luong << "\n";  }  **double** cNhanVienVP::GetLuong(){  **return** luong;  }  **int** cNhanVienVP::isOlder(cNhanVienVP b){  **if**(namSinh < b.namSinh) **return** 1;  **if**(namSinh == b.namSinh && thangSinh < b.thangSinh) **return** 1;  **if**(namSinh == b.namSinh && thangSinh == b.thangSinh && ngaySinh < b.ngaySinh) **return** 1;  **return** 0;  } |

Bảng . - cNhanVienVP.cpp

|  |
| --- |
| #**include** "cNhanVienVP.h"  **class** cListNhanVienVP{  **private**:  vector <cNhanVienVP> NhanVien;  **public**:  **void** **Nhap**();  **void** **Xuat**();  **void** **XuatNVLuongMax**();  **void** **TongLuong**();  **void** **XuatNhanVienTuoiMax**();  **void** **SapXepTangDan**();  }; |

Bảng . - cListNhanVienVP.h

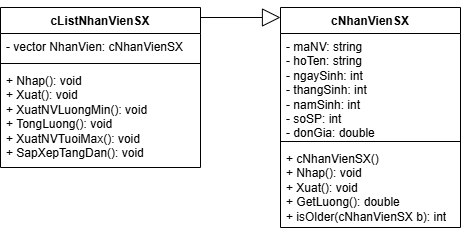
|  |
| --- |
| #**include** "cListNhanVienVP.h"  **void** cListNhanVienVP::Nhap(){  **int** n;  cout << "Ban muon nhap bao nhieu nhan vien: "; cin >> n;  **for**(**int** i = 0; i < n; ++i){  cNhanVienVP x;  x.Nhap();  NhanVien.push\_back(x);  }  }  **void** cListNhanVienVP::Xuat(){  **for**(**int** i = 0; i < NhanVien.size(); ++i){  NhanVien[i].Xuat();  }  }  **void** cListNhanVienVP::XuatNVLuongMax(){  **double** mx = NhanVien[0].GetLuong();  **for**(**int** i = 1; i < NhanVien.size(); ++i){  **if**(NhanVien[i].GetLuong() > mx) mx = NhanVien[i].GetLuong();  }  cout << "Nhan vien co luong cao nhat la: " << "\n";  **for**(**int** i = 0; i < NhanVien.size(); ++i){  **if**(NhanVien[i].GetLuong() == mx) NhanVien[i].Xuat();  }  }  **void** cListNhanVienVP::TongLuong(){  **double** ans = 0;  **for**(**int** i = 0; i < NhanVien.size(); ++i){  ans += NhanVien[i].GetLuong();  }  cout << "Tong luong cong ty phai tra cho nhan vien la: " << ans << "\n";  }  **void** cListNhanVienVP::XuatNhanVienTuoiMax(){  cNhanVienVP mx;  mx = NhanVien[0];  **for**(**int** i = 1; i < NhanVien.size(); ++i){  **if**(NhanVien[i].isOlder(mx)) mx = NhanVien[i];  }  cout << "Nhan vien lon tuoi nhat la: "; mx.Xuat();  }  **void** cListNhanVienVP::SapXepTangDan(){  **for**(**int** i = 0; i < NhanVien.size(); ++i){  **int** mn = i;  **for**(**int** j = i+1; j < NhanVien.size(); ++j){  **if**(NhanVien[j].GetLuong() < NhanVien[mn].GetLuong()) mn = j;  }  swap(NhanVien[mn], NhanVien[i]);  }  cout << "Danh sach nhan vien sau khi sap xep tang dan theo luong la: " << "\n";  Xuat();  } |

Bảng . - cListNhanVien.cpp

|  |
| --- |
| #**include** "cNhanVienVP.h"  #**include** "cListNhanVienVP.h"  **int** **main**(){  cListNhanVienVP NV;  NV.Nhap();  NV.Xuat();  cout << "\n";  NV.XuatNVLuongMax();  cout << "\n";  NV.TongLuong();  cout << "\n";  NV.XuatNhanVienTuoiMax();  cout << "\n";  NV.SapXepTangDan();  cout << "\n";  **return** 0;  } |

Bảng . - main.cpp

# Cài đặt lớp Nhân viên sản xuất (cNhanVienSX) cho phép lưu trữ các thông tin: mã nhân viên, họ tên, ngày sinh, số sản phẩm đã gia công, đơn giá một sản phẩm và các thao tác cần thiết: constructor, get, set, nhập, xuất, tính lương cho nhân viên = số sản phẩm đã gia công \* đơn giá một sản phẩm.

Nội dung class diagram được trình bày như hình 8.

Hình - Class diagram của lớp cNhanVienSX

Nội dung code của chương trình:

|  |
| --- |
| #**ifndef** CNHANVIENSX\_H  #**define** CNHANVIENSX\_H  #**include** <bits/stdc++.h>  **using** **namespace** std;  **class** cNhanVienSX{  **private**:  string maNV;  string hoTen;  **int** ngaySinh;  **int** thangSinh;  **int** namSinh;  **int** soSP;  **double** donGia;  **double** luong;  **public**:  cNhanVienSX();  **void** **Nhap**();  **void** **Xuat**();  **double** **GetLuong**();  **int** **isOlder**(cNhanVienSX b);  };  #**endif** // CNHANVIENSX\_H |

Bảng . – cNhanVienSX.h

|  |
| --- |
| #**include** "cNhanVienSX.h"  cNhanVienSX::cNhanVienSX(){  maNV = "000";  hoTen = "Khong co ten";  ngaySinh = thangSinh = namSinh = 0;  soSP = 0;  donGia = 0;  luong = 0;  }  **void** cNhanVienSX::Nhap(){  cin.ignore();  cout << "Ma nhan vien: "; getline(cin, maNV);  cout << "Ho va ten nhan vien: "; getline(cin, hoTen);  cout << "Ngay sinh: "; cin >> ngaySinh;  cout << "Thang sinh: "; cin >> thangSinh;  cout << "Nam sinh: "; cin >> namSinh;  cout << "So san pham da gia cong: "; cin >> soSP;  cout << "Don gia mot san pham: "; cin >> donGia;  luong = soSP\*donGia;  }  **void** cNhanVienSX::Xuat(){  cout << maNV << " " << hoTen << " " << ngaySinh << "/" << thangSinh << "/" << namSinh << " " << soSP << " " << donGia << " " << luong << "\n";  }  **double** cNhanVienSX::GetLuong(){  **return** luong;  }  **int** cNhanVienSX::isOlder(cNhanVienSX b){  **if**(namSinh < b.namSinh) **return** 1;  **if**(namSinh == b.namSinh && thangSinh < b.thangSinh) **return** 1;  **if**(namSinh == b.namSinh && thangSinh == b.thangSinh && ngaySinh < b.ngaySinh) **return** 1;  **return** 0;  } |

Bảng . - cNhanVienSX.cpp

|  |
| --- |
| #**include** "cNhanVienSX.h"  **class** cListNhanVienSX{  **private**:  vector <cNhanVienSX> NhanVien;  **public**:  **void** **Nhap**();  **void** **Xuat**();  **void** **XuatNVLuongMin**();  **void** **TongLuong**();  **void** **XuatNVTuoiMax**();  **void** **SapXepTangDan**();  }; |

Bảng . - cListNhanVienSX.h

|  |
| --- |
| #**include** "cListNhanVienSX.h"  **void** cListNhanVienSX::Nhap(){  **int** n;  cout << "So nhan vien can nhap la: "; cin >> n;  **for**(**int** i = 0; i < n; ++i){  cNhanVienSX x;  x.Nhap();  NhanVien.push\_back(x);  }  }  **void** cListNhanVienSX::Xuat(){  **for**(**int** i = 0; i < NhanVien.size(); ++i){  NhanVien[i].Xuat();  }  }  **void** cListNhanVienSX::XuatNVLuongMin(){  **double** mn = NhanVien[0].GetLuong();  **for**(**int** i = 1; i < NhanVien.size(); ++i){  **if**(NhanVien[i].GetLuong() < mn) mn = NhanVien[i].GetLuong();  }  cout << "Nhan vien co luong thap nhat la: " << "\n";  **for**(**int** i = 0; i < NhanVien.size(); ++i){  **if**(NhanVien[i].GetLuong() == mn) NhanVien[i].Xuat();  }  }  **void** cListNhanVienSX::TongLuong(){  **double** ans = 0;  **for**(**int** i = 0; i < NhanVien.size(); ++i){  ans = ans + NhanVien[i].GetLuong();  }  cout << "Tong luong cong ty phai tra cho nhan vien la: " << ans << "\n";  }  **void** cListNhanVienSX::XuatNVTuoiMax(){  cNhanVienSX mx;  mx = NhanVien[0];  **for**(**int** i = 1; i < NhanVien.size(); ++i){  **if**(NhanVien[i].isOlder(mx)) mx = NhanVien[i];  }  cout << "Nhan vien lon tuoi nhat la: "; mx.Xuat();  }  **void** cListNhanVienSX::SapXepTangDan(){  **for**(**int** i = 0; i < NhanVien.size(); ++i){  **int** mn = i;  **for**(**int** j = i+1; j < NhanVien.size(); ++j){  **if**(NhanVien[j].GetLuong() < NhanVien[mn].GetLuong()) mn = j;  }  swap(NhanVien[mn], NhanVien[i]);  }  cout << "Danh sach nhan vien sau khi sap xep tang dan theo luong la: " << "\n";  Xuat();  } |

Bảng . - cListNhanVienSX.cpp

|  |
| --- |
| #**include** "cNhanVienSX.h"  #**include** "cListNhanVienSX.h"  **int** **main**(){  cListNhanVienSX NV;  NV.Nhap();  cout << "\n";  NV.Xuat();  cout << "\n";  NV.XuatNVLuongMin();  cout << "\n";  NV.TongLuong();  cout << "\n";  NV.XuatNVTuoiMax();  cout << "\n";  NV.SapXepTangDan();  cout << "\n";  **return** 0;  } |

Bảng . - main.cpp

# TỔNG HỢP ĐƯỜNG LINK CODE CÁC BÀI

Tổng hợp các code tại drive sau: <https://drive.google.com/drive/folders/1Ays5Ek2kxM90vhBkeG5UisNavSESweIU?usp=drive_link>

**- HẾT -**