**TRƯỜNG ĐẠI HỌC HÀNG HẢI VIỆT NAM  
KHOA CÔNG NGHỆ THÔNG TIN**

**-----\*\*\*-----**

****

**BÁO CÁO BÀI TẬP LỚN**

**HỌC PHẦN “LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG”**

***Đề tài:***

***TÌM HIỂU CÁCH XÂY DỰNG LỚP VÀ CÁC PHƯƠNG THỨC***

***LIÊN QUAN***

***GVHD: ThS. Nguyễn Hạnh Phúc***

***Sinh viên thực hiện: Ngô Đăng Hoàng***

***Bùi Quang Hiển***

***Hải Phòng, tháng 10 năm 2016***

**TRƯỜNG ĐẠI HỌC HÀNG HẢI VIỆT NAM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN KHOA HỌC MÁY TÍNH**

**-----\*\*\*-----**

**BÀI TẬP LỚN**

**HỌC PHẦN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

**Mã đề tài: 08**

1. **Tên đề tài**

* ***Xây dựng lớp CMatrix lưu thông tin về ma trận gồm số hàng, số cột và mảng***  
  ***động là các số thực trong ma trận. Lớp này có các phương thức: khởi tạo,***  
  ***nhập, xuất, khởi tạo sao chép, hủy bỏ, Max (tìm vị trí phần tử lớn nhất ma***  
  ***trận) và các toán tử + (cộng 2 ma trận), \* (nhân 2 ma trận)***

1. **Mục đích**

Tìm hiểu cách xây dựng lớp, các phương thức trong lớp và lưu thông tin về ma trận, mảng động, các phép toán liên quan đến ma trận.

1. **Công việc cần thực hiện**

* Tìm hiểu cách xây dựng lớp
* Tìm hiểu các phương thức trong lớp
* Ứng dụng xây dựng một báo cáo (sinh viên tự đề xuất)
* Làm báo cáo bài tập lớn
* Bảo vệ bài tập lớn

1. **Yêu cầu**

* Kết quả làm bài tập lớn: Báo cáo bài tập lớn
* Báo cáo bài tập lớn phải được trình bày theo mẫu quy định (kèm theo), báo cáo có thể kết xuất thành tệp định dạng PDF và nộp qua email (không bắt buộc phải in ấn)
* Hạn nộp báo cáo bài tập lớn: 24/5/2017

1. **Tài liệu tham khảo**
   * [Giáo trình bài giảng Lập Trình Hướng Đối Tượng](http://www.amazon.com/Paul-Turley/e/B001IGJNGC/ref=ntt_athr_dp_pel_1)…

***Hải Phòng, ngày 30 tháng 10 năm 2016***

NGƯỜI HƯỚNG DẪN

# MỤC LỤC

# DANH MỤC CÁC HÌNH VẼ, BẢNG BIỂU

**DANH MỤC CÁC TỪ VIẾT TẮT**

**GIỚI THIỆU**

**CHƯƠNG 1. ĐỐI TƯỢNG VÀ LỚP**

* 1. Định nghĩa đối tượng và lớp
  2. Khai báo lớp đối tượng
  3. Hàm tạo, hàm sao chép và hàm hủy
  4. Hàm bạn và lớp bạn
  5. Chồng toàn tử (overload operator)

**CHƯƠNG 2. ỨNG DỤNG**

*Bài tập*: Xây dựng lớp CMatrix lưu thông tin về ma trận gồm số hàng, số cột và mảng động là các số thực trong ma trận. Lớp này có các phương thức: khởi tạo, nhập, xuất, khởi tạo sao chép, hủy bỏ, Max (tìm vị trí phần tử lớn nhất ma  
trận) và các toán tử + (cộng 2 ma trận), \* (nhân 2 ma trận).

# 

**CHƯƠNG 1**

**ĐỐI TƯỢNG VÀ LỚP**

* 1. Định nghĩa dối tượng và lớp
* Đối tượng :
  + là một thực thể trong thế giới thực, mỗi đối tượng có đặc điểm và hành động của riêng nó. Đặc điểm của đối tượng được mô tả thông qua thuộc tính và thành phần dữ liệu
  + là một biến có kiểu lớp.

VD:

* Lớp:
  + là tập các đối tượng cùng đặc điểm cùng hành động
  + là sự trừu tượng hóa dữ liệu và phương thức
  + là một kiểu dữ liệu có cấu trúc.

1. Khai báo lớp và đối tượng

2.1. Khai báo lớp

- Khai báo lớp ta sử dụng từ khóa **class** :

**class** **tên\_lớp[:<kiểu thừa kế>lop\_cha]{**

**// Khai báo các thành phần dữ liệu (thuộc tính)**

**// Khai báo các phương thức (hàm)**

**};**

- Chi tiết hơn ta có khai báo lớp như sau :

**class** **tên\_lớp[:<kiểu thừa kế>lop\_cha]{**

**private :**

// Khai báo các thành phần dữ liệu (thuộc tính) riêng

// Khai báo các phương thức (hàm) riêng

**protected:**

// Khai báo các thành phần dữ liệu (thuộc tính) được bảo vệ

// Khai báo các phương thức (hàm) được bảo vệ

**public:**

// Khai báo các thành phần dữ liệu (thuộc tính) chung

// Khai báo các phương thức (hàm) chung

**};**

**- Trong đó :**

* class, private, public, protected… : là các từ khóa dùng để khai báo lớp.
* Tên\_Lớp : là do người lập trình đặt.
* [:<kiểu thừa kế>lop\_cha] : chỉ có thể xảy ra khi thừa kế.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NHÃN | Trong lớp | Ngoài lớp | Bạn bè | Con |
| Private | X |  | X |  |
| Protected | X |  | X | X |
| Public | X | X | X | X |

* Private, Protected, Public : không nhất thiếtđặt đúng thứ tự; có thể có hoặc không có trong lớp; Private thường đi với dữ liệu; Public thường đi với phương thức; Protected thừa kế. Nếu dữ liệu và phương thức không có nhãn thì mặc định chúng có nhãn Ptivate.

2.2. Khai báo đói tượng

- Cấu trúc khai báo đối tượng: **tên\_lớp tên\_đtượng;**

VD:

SV a; SV \*p; SV &q = a;

**-** Truy cập:

* a.ten : truy cập vào trường tên của đối tượng a.
* a.nhap() : truy cập vào phương thức “nhap” của đối tượng a.
* p->msv : truy cập vào trường msv của con trỏ p.
* p->in : truy cập vào phương thức “in” của con trỏ p
* q.kt() (tương tự như a.kt()) truy cập vào phương thức “kt” của đối tượng q.

VD: Thông tin của sinh viên: Tên và mã sinh viên. Hãy xây dựng lớp SV có thông tin trên và có phương thức khởi tạo, nhập, xuất.

**class** SV{

**private:**

string ten;

int msv;

**public:**

SV(){

ten = ””; msv = 0;

}

void nhap();

void in();

};

1. Hàm tạo, hàm sao chép và hàm hủy

3.1. Hàm tạo (Hàm khởi tạo, cấu tử)

* Hàm tạo là hàm luôn luôn tồn tại trong lớp, dùng để khởi tạo giá trị ban đầu cho các thành phần dữ liệu ban đầu có các đối tượng trong lớp.

- Đặc điểm:

* Có tên trùng với tên lớp.
* Có thuộc tính public.
* Không có kiểu trả về.
* Có thể có hoặc không có tham số (trong lớp có thể có nhiều hàm tạo).
* Hàm tạo tự động được gọi khi có một đối tượng khai báo.
* Nếu không xây dựng tường minh hàm tạo thì chương trình sẽ tự sinh ra một hàm tạo không có nội dung, không có tham số. Hàm không có tham số là hàm mặc định.

- Cấu trúc hàm tạo:

**Tên\_lớp**( // tham số truyền vào nếu có){

// nội dung

}

VD: Xây dựng lớp PS với 2 hàm tạo, hàm tạo mặc định và hàm tạo tham số.

**class** PS{

**private:**

int ts,ms;

**public:**

PS(){ ts = 1; ms = 2; } // hàm tạo mặc định

PS(int t, int m){ ts = t; ms = m } // hàm tạo tham số

void nhap();

void in();

};

3.2. Hàm sao chép

**Tên\_lớp**( // Tham chiếu đến lớp ){

// nội dung

}

***Chú ý:*** Với các lớp không có thành phần con trỏ thì không cần cấu tử sao chép tường minh; Chương trình tự sinh ra một cấu tử sao chép không có nội dung, và được gọi là sao chép mặc định.

VD:

**class** Ex1{

**private:**

int a;

int \*b;

**public:**

Ex1(){ a = 1; b = NULL; } // Hàm tạo mặc định

Ex1(Ex1 &p){ // Hàm sao chép

a = p.a;

b = new int;

b = p.b;

}

// …

};

3.3. Hủy tử

- Hủy tử là phương thức luôn luôn tồn tại trong lớp có tác dụng hủy bỏ đối tượng khi đối tượng ra khỏi tầm vực chứa đối tượng đó.(thu hồi bộ nhớ đóng file, dọn dẹp file rác,…).

- Đặc điểm:

* Tên trùng tên lớp và có dấu “~” ở phía trước.
* Có thuộc tính public.
* Không có tham số (Có duy nhất một hàm hủy trong lớp).
* Không có kiểu trả về.
* Hàm tự động được gọi khi một đối tượng ra khỏi khu vực của nó.
* Nếu không xây dựng hàm hủy tường minh thì chương trình tự sinh ra một hàm hủy không có nội dung

***Chú ý:*** Với các lớp không có thành phần con trỏ thì không cần hàm hủy tường minh.

VD:

**class** Ex1{

**private:**

int a;

int \*b;

**public:**

Ex1(){

a = 1; b = NULL;

}

~Ex(){ // Hàm hủy

Delete []b;

}

// …

};

1. Các hàm bạn và lớp bạn

- Các thành phần private chỉ được truy cập từ các phương thức trong lớp, trong nhiều trường hợp cần truy cập các thành phần trên từ bên ngoài nên ta sử dụng **friend.**

**-** Phân loại:

* 1. Hàm tự do là bạn của lớp (một hoặc nhiều lớp).
  2. Hàm của lớp A là bạn của lớp B.
  3. Mọi hàm của lớp A là bạn của lớp B (lớp A là bạn của lớp B).

- Hàm bạn là hàm không thuộc lớp, nhưng có thể truy cập vào mọi thành phần của lớp đó. Hàm bạn được khai báo vời từ khóa **friend.**

**-** Không có bạn bè 2 chiều.

- Khai báo:

* Trường hợp 1: Hàm tự do “fabc()” là bạn của lớp A.

class A{

// nội dung của A

friend kiểu\_fabc fabc(tham số nếu có);

};

kiểu\_fabc fabc(tham số nếu có){

// nội dung fabc

}

* Trường hợp 2: Hàm của lớp A là bạn với lớp B.

class A;

class B{

// nội dung của B

friend kiểu\_fA1 A::fA1(tham số nếu có);

};

class A{

//nội dung của A

kiểu\_fA1 fA1(tham số nếu có);

kiểu\_fA2 fA1(tham số nếu có);

};

* Trường hợp 3: Lớp A là bạn với lớp B.

class A;

class B{

//nội dung của B

friend class A;

};

class A{

//nội dung của A

…

};

**CHƯƠNG 2**

**ỨNG DỤNG**

*Bài tập*: Xây dựng lớp CMatrix lưu thông tin về ma trận gồm số hàng, số cột và mảng động là các số thực trong ma trận. Lớp này có các phương thức: khởi tạo, nhập, xuất, khởi tạo sao chép, hủy bỏ, Max (tìm vị trí phần tử lớn nhất ma  
trận) và các toán tử + (cộng 2 ma trận), \* (nhân 2 ma trận).

#include<iostream>

#include<iomanip>

using namespace std;

class CMatrix

{

int hang,cot;

float \*\*a;

float \* MangDong;

public:

CMatrix();

CMatrix(int p, int q);

CMatrix(CMatrix &x);

~CMatrix();

void nhap();

void xuat();

void dayso();

void max();

int ktrCong(CMatrix &x);

int ktrNhan(CMatrix &x);

friend CMatrix operator+(CMatrix a, CMatrix b);

friend CMatrix operator\*(CMatrix a, CMatrix b);

};

void CMatrix::nhap(){

cout<<"Nhap so hang: ";cin>>hang;

cout<<"Nhap so cot: ";cin>>cot;

a= new float \*[hang];

for (int i=0;i<hang;i++)

a[i]= new float [cot];

for (int i=0;i<hang;i++){

for (int j=0;j<cot;j++)

{

cout<<"a["<<i<<j<<"]: ";

cin>>a[i][j];

}

}

}

CMatrix::~CMatrix(){

delete []MangDong;

for (int i=0;i<hang;i++){

delete a[i];

}

delete a;

}

CMatrix::CMatrix(CMatrix &x){

hang=x.hang;

cot=x.cot;

MangDong = new float [hang\*cot];

a = new float \*[hang];

for(int i=0;i<hang;i++){

a[i] = new float[cot];

}

for(int i=0;i<hang;i++){

for(int j=0;j<cot;j++){

a[i][j] = x.a[i][j];

}

}

}

CMatrix::CMatrix(int p, int q){

hang=p;

cot=q;

MangDong = new float[hang \* cot];

a = new float \*[hang];

for(int i=0;i<hang;i++){

a[i] = new float [cot];

}

}

CMatrix::CMatrix(){

MangDong = NULL;

hang =0;

cot =0;

a = new float \*[hang];

for(int i=0;i<hang;i++){

a[i] = new float [cot];

}

}

**// xuat du lieu ra man hinh.**

void CMatrix::xuat()

{

for(int i=0;i<hang;i++){

cout <<"|";

for(int j=0;j<cot;j++){

cout <<setw(5)<< a[i][j]<<" |";

}

cout << endl;

}

}

void CMatrix::dayso()

{

int t=0;

MangDong = new float [hang\*cot];

for(int i=0;i<hang;i++){

for(int j=0;j<cot;j++){

MangDong[t] = a[i][j];

t++;

}

}

for(int k=0;k<t;k++){

cout <<MangDong[k]<<", ";

}

}

void CMatrix::max()

{

float Max;

int i,j;

Max = a[0][0];

for(int i=0;i<hang;i++){

for(int j=0;j<cot;j++){

if(Max < a[i][j]){

Max = a[i][j];

}

}

}

cout << "\nMax: " << Max << endl;

cout<<"Vi tri cua Max la: ";

for(i=0;i<hang;i++){

for(j=0;j<cot;j++){

if(Max == a[i][j]){

cout<<"A["<<i<<"]["<<j<<"]"<<setw(5);

}

}

}

}

**// kiem tra 2 ma tran co CONG duoc voi nhau hay khong.**

int CMatrix::ktrCong(CMatrix &x){

if(hang == x.hang && cot == x.cot) return 1;

return 0;

}

**// kiem tra 2 ma tran co NHAN duoc voi nhau hay khong.**

int CMatrix::ktrNhan(CMatrix &y){

if(cot == y.hang) return 1;

return 0;

}

**// CONG 2 ma tran**

CMatrix operator+(CMatrix d, CMatrix e){

CMatrix c=d;

for(int i=0;i<d.hang;i++){

for(int j=0;j<d.cot;j++){

c.a[i][j] = d.a[i][j] + e.a[i][j];

}

}

return c;

}

**// NHAN 2 ma tran**

CMatrix operator\*(CMatrix d, CMatrix e){

CMatrix c;

for(int i=0;i<d.hang;i++){

for(int j=0;j<d.cot;j++){

CMatrix c;

for(int t=0;t<c.hang;t++){

c.a[i][j] += d.a[i][t] \* e.a[t][j];

}

}

}

return c;

}

**// HAM chinh**

int main()

{

int Control;

CMatrix d,e;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl

<<" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* " << endl

<< " CHUONG TRINH NHAP VA TINH TOAN TREN MA TRAN NHOM 08" << endl

<< "START" << endl;

cout<<" 1. Nhap ma tran."<< endl

<<" 2. In day so thuc trong ma tran." <<endl

<<" 3. Tim Max cua ma tran." <<endl

<<" 4. Cong 2 ma tran." <<endl

<<" 5. Nhan 2 ma tran." <<endl

<<" 6. Thoat." <<endl;

do

{

cout << endl<<"---------------------------------" << endl

<<"Lua chon tinh nang cua chuong trinh: ";

cin >> Control;

switch(Control)

{

case 1:

{

cout<< "Nhap ma tran D :" << endl;

d.nhap();

cout << "Ma tran ban vua nhap la:" << endl;

d.xuat();

break;

}

case 2:

{

cout <<"Cac so thuc trong ma tran: "<< endl;

d.dayso();

break;

}

case 3:

{

d.max();

break;

}

case 4:

{

cout<<"CHUONG TRINH CONG 2 MA TRAN!!!"<<endl;

cout<<"Nhap ma tran E:"<<endl;

e.nhap();

if(d.ktrCo

ng(e)==1){

d.xuat();

cout<<" + \n";

e.xuat();

cout<<" = \n";

(d+e).xuat();

}

else cout<<"Hai ma tran khong cung cap!!!";

break;

}

case 5:

{

cout<<"CHUONG TRINH NHAN 2 MA TRAN!!!"<<endl;

cout<<"Nhap ma tran E:"<<endl;

e.nhap();

if(d.ktrNhan(e) == 1){

d.xuat();

cout<<" \* \n";

e.xuat();

cout<<" = \n";

(d\*e).xuat();

}

else cout<<"Khong the nhan 2 ma tran!!!";

break;

}

case 6:

{

return 0;

break;

}

default :

{

cout<<"<!> Khong tim thay lua chon cua ban ! Nhap lua chon tu 1 -> 7";

break;

}

}

}while(Control!=6);

return 0;

}