

Lập trình hướng đối tượng

BÀI 2

QUAN HỆ KẾT TẬP VÀ PHẠM VI TRUY CẬP

Nội dung

1. Phương thức getter(), setter()
2. Quan hệ kết tập (lồng nhau)
3. Hàm bạn, lớp bạn.
4. Mảng đối tượng

2.1. Phương thức getter(), setter()

- Phương thức getter(): Để trả về giá trị thuộc tính của đối tượng.


```
class Nguoi{  
    private:  
        char hoTen[30];  
        char ngaySinh[11];  
        float chieuCao;  
    public:  
        ...  
        float getChieuCao() {  
            return chieuCao;  
        }  
};
```

- Ví dụ:

Phương thức getter(), setter()

- Phương thức getter(): Sử dụng khi muốn lấy giá trị của thuộc tính **private** của đối tượng (từ bên ngoài lớp).

```
class Ngươi{  
    ...  
    public:  
        ...  
        float getChieuCao() {  
            return chieuCao;  
        }  
};  
  
int main() {  
    Ngươi n;  
    cout<<"Chieu cao la: "<<n.getChieuCao();  
}
```



Phương thức getter(), setter()

- Phương thức setter(): Để thiết lập giá trị mới cho thuộc tính của đối tượng.

- Ví dụ:

```
class Nguoi{  
    private:  
        char hoTen[30];  
        char ngaySinh[11];  
        float chieuCao;  
    public:  
        ...  
        void setChieuCao(float t) {  
            chieuCao = t;  
        }  
};
```

Phương thức getter(), setter()

- Phương thức setter(): Sử dụng khi muốn thiết lập giá trị mới cho thuộc tính **private** của đối tượng (từ bên ngoài lớp).

```
class Ngnoi{
    ...
    public:
        ...
        void setChieuCao(float t) {
            chieuCao = t;
        }
};

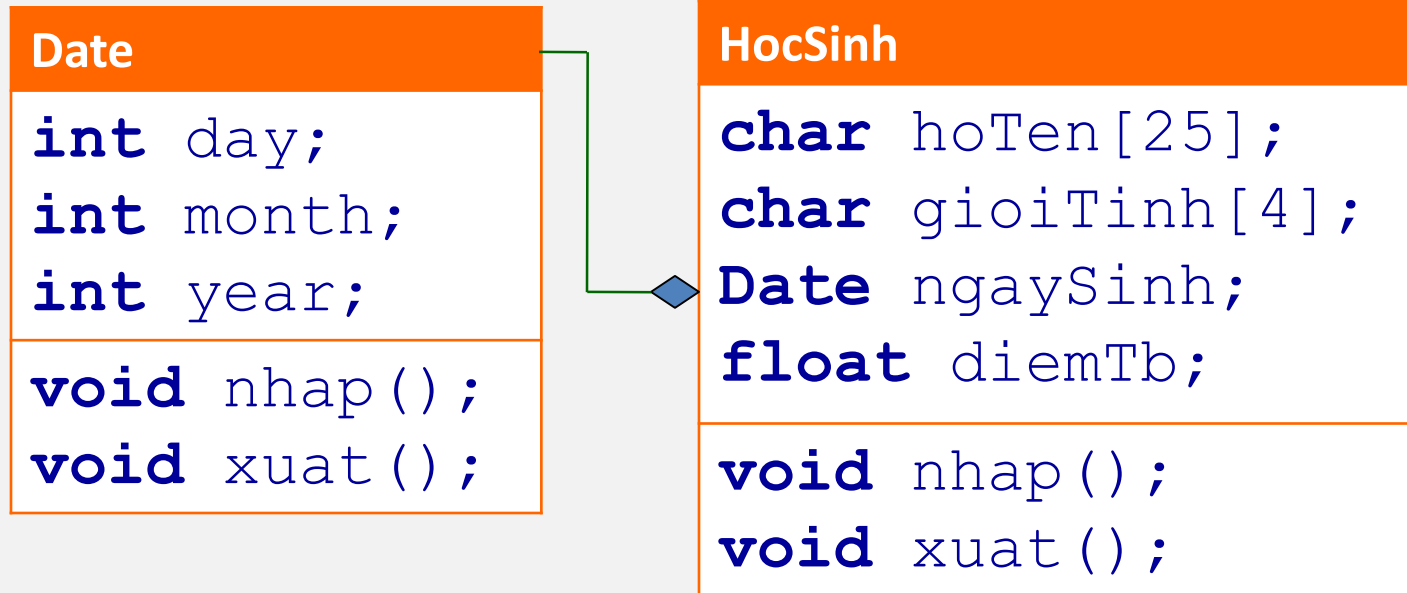
int main() {
    Ngnoi n;
    n.setChieuCao(1.75); //n.chieuCao = 1.75
}
```

Phương thức getter(), setter() – Ví dụ

- **Chương trình giải quyết bài toán:**
 - Tạo ra một đối tượng học sinh và khởi tạo các thông tin cho đối tượng gồm: Họ tên, ngày, tháng, năm sinh, giới tính, điểm trung bình, xếp loại đạo đức.
 - Hiển thị thông tin của học sinh ra màn hình.
- **Chương trình sử dụng phương thức setter(), getter().**

2.2. Quan hệ kết tập

- Khi thuộc tính của một lớp là một đối tượng của lớp khác.
- Sơ đồ lớp cho quan hệ kết tập.



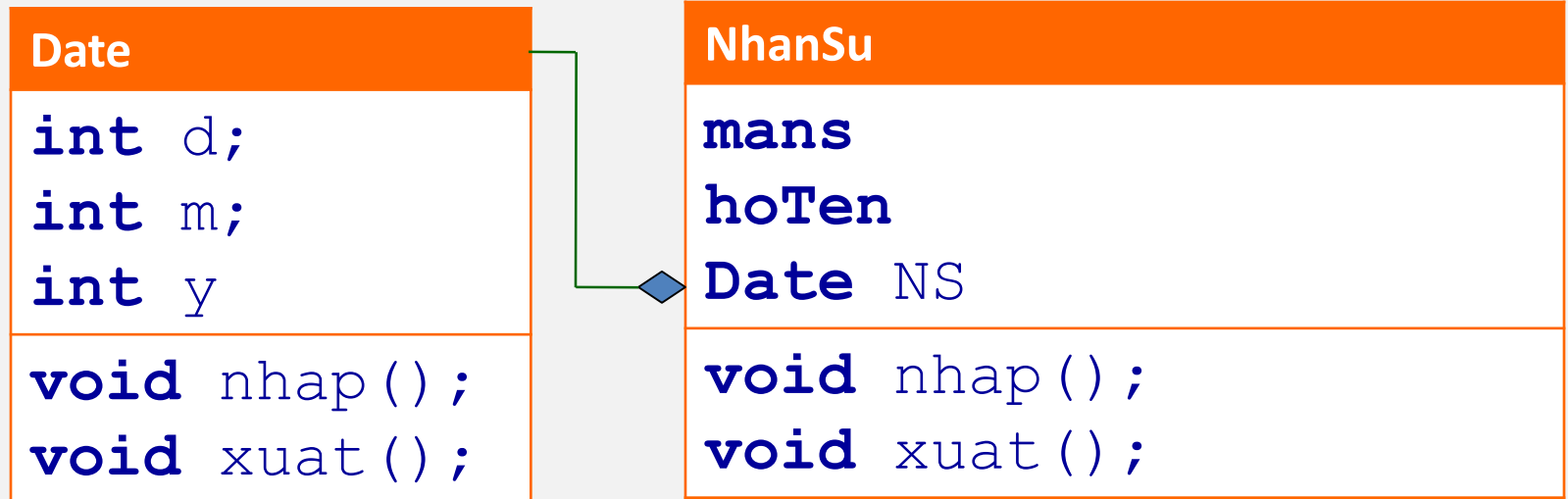
Quan hệ kết tập (tt)

```
class Date {  
    private:  
        int day, month, year;  
    ...  
};  
class HocSinh {  
    private:  
        char hoTen[30];  
        char gioiTinh[4];  
        Date ngaySinh;  
        float diemTb;  
    public:  
        ...  
};
```

- Thuộc tính ngaySinh của lớp HocSinh là một đối tượng của lớp Date.
- Cần chú ý việc truy xuất vào các thuộc tính **private** của lớp Date.

Quan hệ kết tập (tt) – Ví dụ 1

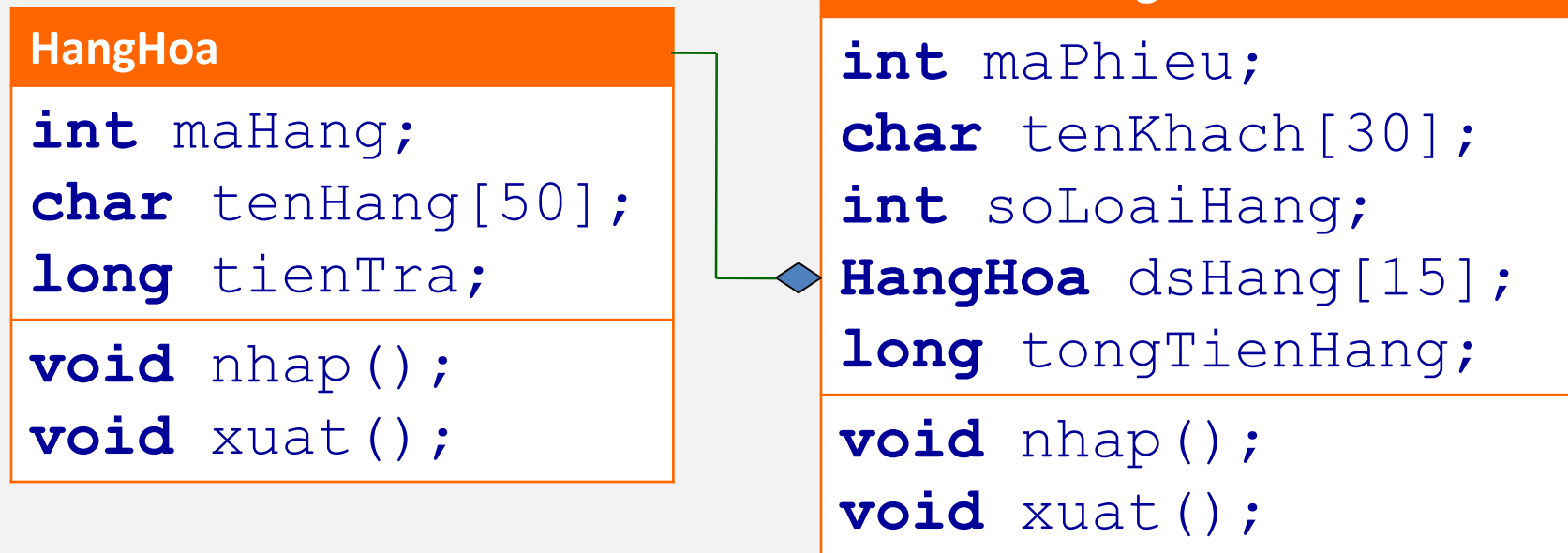
- Cài đặt chương trình theo sơ đồ lớp dưới đây



- Yêu cầu: Cài đặt hàm main nhập vào một nhân sự x, in thông tin của nhân sự ra màn hình.

Quan hệ kết tập (tt) – Ví dụ 2

- Cài đặt chương trình theo sơ đồ lớp dưới đây



- Yêu cầu: Nhập thông tin cho một phiếu mua hàng, hiển thị thông tin phiếu mua hàng ra màn hình.

2.3. Hàm bạn – Lớp bạn

- Hàm bạn của một lớp
 - Để truy xuất vào các thành phần riêng tư của lớp
- Khai báo hàm bạn

```
class A{  
    private:  
        //Khai báo các thuộc tính của lớp A  
    public:  
        //Khai báo các phương thức của lớp A  
    //Khai báo hai hàm bạn f1 và f2  
    friend <nguyên mẫu của hàm f1>;  
    friend <nguyên mẫu của hàm f2>;  
};  
  
// Xây dựng các hàm f1 và f2 bên ngoài lớp A
```

Tính chất – đặc điểm của hàm bạn

- Hàm không nằm trong phạm vi của lớp mà nó đã được khai báo là friend.
- Nó không thể được gọi bằng cách sử dụng đối tượng vì nó không nằm trong phạm vi của lớp đó.
- Nó có thể được gọi như một hàm bình thường mà không cần sử dụng đối tượng.
- **Nó không thể truy cập trực tiếp vào tên thành viên và phải sử dụng tên đối tượng và dấu chấm toán tử với tên thành viên.**
- Nó có thể được khai báo trong phần private hoặc public.
- (Trong thân hàm bạn của một lớp có thể truy nhập tới các thuộc tính của các đối tượng thuộc lớp này. Đây là khác nhau duy nhất giữa hàm bạn và hàm thông thường.
 - + Hàm bạn không phải là phương thức của một lớp, lời gọi của hàm bạn giống như lời gọi của hàm thông thường.
-)

2.3.1. Hàm bạn – Ví dụ

```
class A{
    private:
        int a, b;
    public:
        void set(){
            a = 2; b = 3;
        }
    // Khai báo hàm tinh() là bạn của A
        friend void tinh(A dt);
};
void tinh(A dt){
    cout<<"Dien tich: "<<dt.a*dt.b;
}
int main(){
    A a;
    a.set();  tinh(a);
}
```

2.3.2. Lớp bạn

Khi tất cả các phương thức của một lớp là bạn của một lớp khác, thì lớp của các phương thức đó cũng trở thành lớp bạn của lớp kia. Muốn khai báo một lớp B là lớp bạn của lớp A.

Lớp friend được xây dựng để khắc phục điểm yếu **lớp dẫn xuất không thể truy cập tới các biến private của lớp cơ sở**

a. Định nghĩa:

Một friend có thể là một hàm, một mẫu hàm, hoặc hàm thành viên, hoặc một lớp hoặc một mẫu lớp, trong trường hợp này, toàn bộ lớp và tất cả thành viên của nó là friend.

Hàm friend trong C++ của một lớp được định nghĩa bên ngoài phạm vi lớp đó, nhưng nó có quyền truy cập tất cả thành viên private và protected của lớp đó. Ngay cả khi các nguyên mẫu cho hàm friend xuất hiện trong định nghĩa lớp, thì các hàm friend không là các hàm thành viên.

2.3.2. Lớp bạn

b.Tính chất:

- Friend của một class có thể là thành viên của 1 class khác
- Friend của 1 class có thể là thành viên của class khác hoặc tất cả các class trong cùng 1 chương trình. Như là 1 GLOBAL FRIEND
- Friend có thể access private hoặc protected của class được khai báo là Friend.
- Friends không phải là một thành viên vì vậy không có con trỏ "this"
- Friend có thể khai báo ở bất cứ đâu (public, protected or private section) trong một class.

2.3.2. Lớp bạn

c.Khai báo tuần tự theo sau:

Khai báo khuôn mẫu lớp B:

class B;

Định nghĩa lớp A, với khai báo B là lớp bạn:

class A

{

...// Khai báo các thành phần của lớp A

// Khai báo lớp bạn B

friend class B;

};

Định nghĩa chi tiết lớp B:

class B

{ ... // Khai báo các thành phần của lớp B };

Lưu ý:

- Trong trường hợp này, lớp B là lớp bạn của lớp A nhưng không có nghĩa là lớp A cũng là bạn của lớp B: tất cả các phương thức của lớp B có thể truy nhập các thành phần private của lớp A (thông qua các đối tượng có kiểu lớp A) nhưng các phương thức của lớp A lại không thể truy nhập đến các thành phần private của lớp B.
- Muốn các phương thức của lớp A cũng truy nhập được đến các thành phần private của lớp B, thì phải khai báo thêm là lớp A cũng là lớp bạn của lớp B.

2.3.2. Lớp bạn

- Lớp bạn của một lớp: Để các phương thức của lớp có thể truy xuất vào các thành phần riêng tư của lớp khác.
- Ví dụ:

```
class A{  
    private:  
        int a, b;  
    public:  
        void nhap() {  
            cout<<"Nhap a, b: ";  
            cin>>a>>b;  
        }  
};
```

```
class B{  
    private:  
        int c;  
    public:  
        void tinh(A dt) {  
            cout<<"dien tich:";  
            cout<<dt.a * dt.b;  
        }  
};
```

- Trong trường hợp này chương trình báo lỗi vì **tinh()** không phải là phương thức của lớp A.

Lớp bạn (tt)

- Khi đó lớp B phải là bạn của lớp A

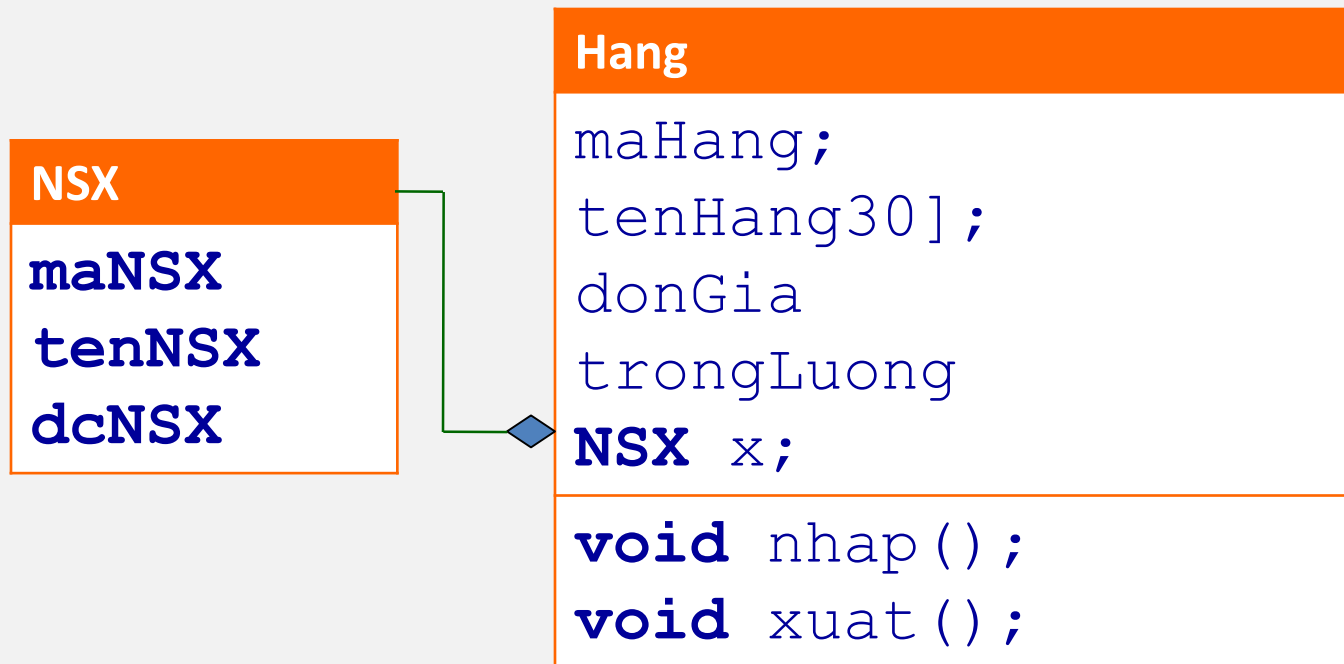
```
class A {  
    private:  
        int a, b;  
    public:  
        void nhap() {  
            cout<<"Nhap a, b ";  
            cin>>a>>b;  
        }  
    //KB B la ban cua lop A  
    friend class B;  
};
```

```
class B{  
    public:  
        void tinh(A dt){  
            cout<<"D.tich: ";  
            cout<< dt.a * dt.b;  
        }  
};  
  
int main() {  
    A d1;  
    B d2;  
    d1.nhap();  
    d2.tinh(d1);  
}
```

Chương trình này chạy tốt!

Lớp bạn (tt) – Ví dụ

- Cài đặt chương trình theo sơ đồ lớp dưới đây



- Yêu cầu: Nhập thông tin của một mặt hàng, hiển thị thông tin của mặt hàng đó ra màn hình.

2.4. Mảng đối tượng

- Mảng đối tượng được tạo ra để lưu trữ danh sách đối tượng trong chương trình.
- Khai báo mảng đối tượng.

```
<Tên_Lớp> <tên_mảng><[kích_thước]>;
```

- Mảng đối tượng cũng giống các mảng khác, với mỗi phần tử mảng là (dùng để chứa) một đối tượng.
- Ví dụ:

```
CanBo cb[20];
```

```
//Mảng chứa 20 đối tượng cán bộ
```

Mảng đối tượng (tt) – Ví dụ

- Cài đặt chương trình thực hiện các yêu cầu:
 - Cài đặt lớp SinhVien (sinh viên) bao gồm các thuộc tính: Mã sinh viên, họ và tên, ngày sinh, giới tính, ngành học, điểm tổng kết và các phương thức cần thiết.
 - Cài đặt các chức năng:
 - Nhập vào một danh sách mới gồm n sinh viên.
 - Hiển thị danh sách n sinh viên vừa nhập ra màn hình.
 - Hiển thị ra màn hình thông tin của những sinh viên có điểm tổng kết cao nhất.

2.5. Con trỏ đối tượng

- Khai báo con trỏ đối tượng

`<Tên_Lớp> *<tên_con_trỏ>;`

- Trong đó:

- `<Tên_Lớp>`: Lớp chứa đối tượng mà con trỏ sẽ trỏ tới.
- `<tên_con_trỏ>`: đặt theo quy ước đặt tên trong C++.

- Ví dụ: `Nguoì nguoi, *p;`

- Cho con trỏ `p` trỏ tới đối tượng `nguoi` bằng lệnh gán:

`p = &nguoi;`

- Khi con trỏ `p` đã trỏ tới `nguoi`, có thể dùng con trỏ `p` để truy xuất đến các thành phần của đối tượng `nguoi`.
- Khi đó ta viết:

`<tên_con_trỏ>-><tên_thành_phần>;`

Con trỏ đối tượng (tt)

- Hai cách viết như sau là tương đương

Đối tượng nguoi thuộc lớp Nguoi	Con trỏ p chứa địa chỉ của nguoi
nguoi .hoTen	p-> hoTen
nguoi .ngaySinh	p-> ngaySinh
nguoi .nhap()	p-> nhap()
...	...

Con trỏ đối tượng (tt) – Ví dụ

```
class Ngnoi{
    private:
        char hoTen[30];
        int tuoi;
        char que[60];
    public:
        void nhap() {...}
        void xuat() {...}
};

int main(){
    Ngnoi ngnoi, *p;
    p = &ngnoi;
    p->nhap();
    p->xuat();
}
```

Con trỏ đối tượng (tt) – Con trỏ this

- **this** – con trỏ ngầm định: Là một con trỏ, được tự động tạo ra và luôn trỏ vào đối tượng hiện thời (đối tượng đang thực thi trong chương trình).
- Xét ví dụ:

```
class HìnhChuNhat {  
    private:  
        double chieuDai, chieuRong;  
    public:  
        void nhap() {  
            cout<<"Chieu dai: "; cin>>chieuDai;  
            cout<<"Chieu rong: "; cin>>chieuRong;  
        }  
};
```

Con trỏ đối tượng (tt) – Con trỏ this

- Thực chất sẽ viết là

```
class HìnhChuNhat {  
    private:  
        double chieuDai, chieuRong;  
    public:  
        void nhap() {  
            cout<<"Chieu dai: ";  
            cin>>this->chieuDai;  
            cout<<"Chieu rong: ";  
            cin>>this->chieuRong;  
        }  
};
```

2.5. Bài tập - 01

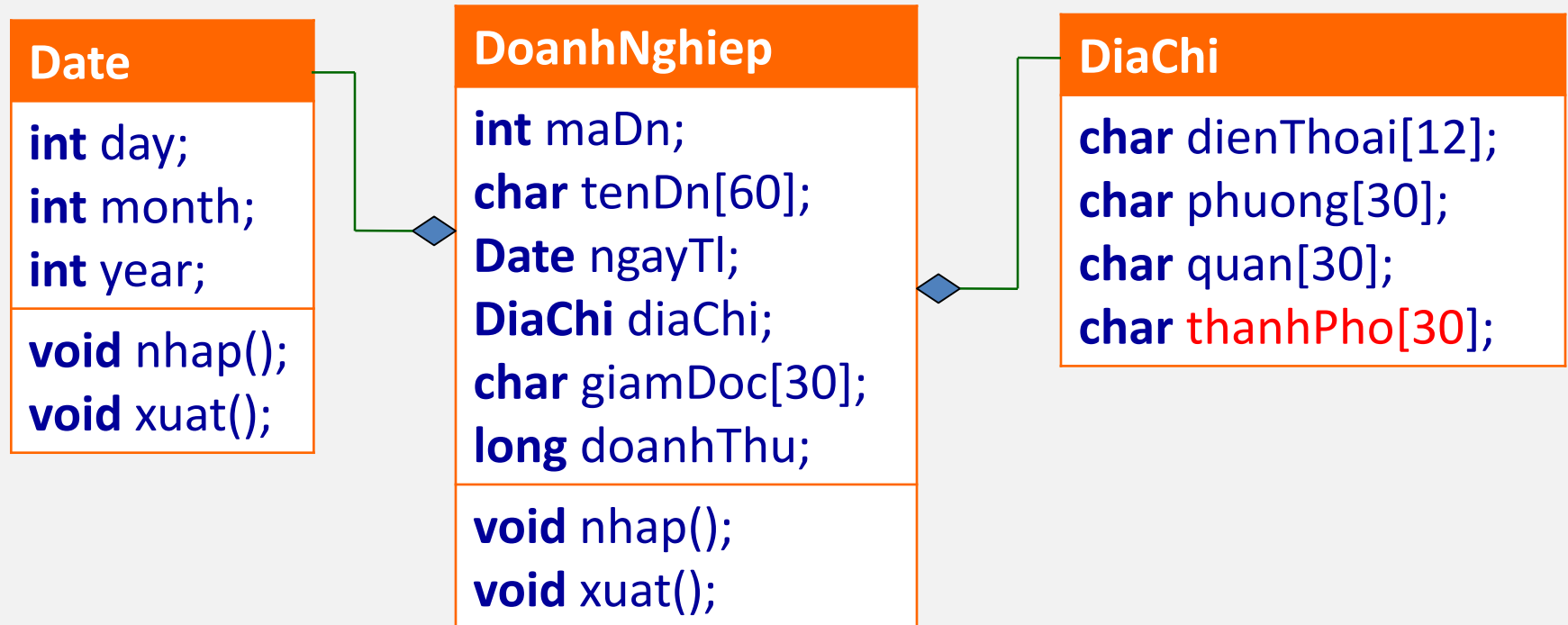
- Cài đặt chương trình thực hiện các yêu cầu:
 - Thiết kế một lớp HoaDon (Hoá đơn xuất hàng) bao gồm các thuộc tính: Mã hóa đơn, đơn vị nhận hàng, số tiền, người thanh toán, người nhận, ngày thanh toán, danh sách các hàng hóa (mỗi hàng hóa gồm mã hàng, tên hàng, số lượng, giá bán) và các phương thức cần thiết.
 - Cài đặt các chức năng:
 - Nhập vào thông tin của một hóa đơn,.
 - Xuất thông tin của hoá đơn lên màn hình.
 - Cho biết tổng tiền của hóa đơn.

Bài tập (tt) - 02

- Cài đặt chương trình thực hiện các yêu cầu
 - Cài đặt lớp XeHoi (xe hơi) gồm các thuộc tính: Nhãn hiệu, **hãng sản xuất**, kiểu dáng, màu sơn, năm sản xuất, xuất xứ, giá bán và các phương thức cần thiết.
 - Nhập vào một danh sách n xe hơi.
 - Hiển thị danh sách ra màn hình.
 - Hiển thị ra màn hình những xe hơi **của hãng “Toyota”**.
 - Sắp xếp danh sách theo chiều tăng dần của **giá bán**, in kết quả lên màn hình.

Bài tập (tt) - 03

- Cài đặt chương trình theo sơ đồ lớp sau:

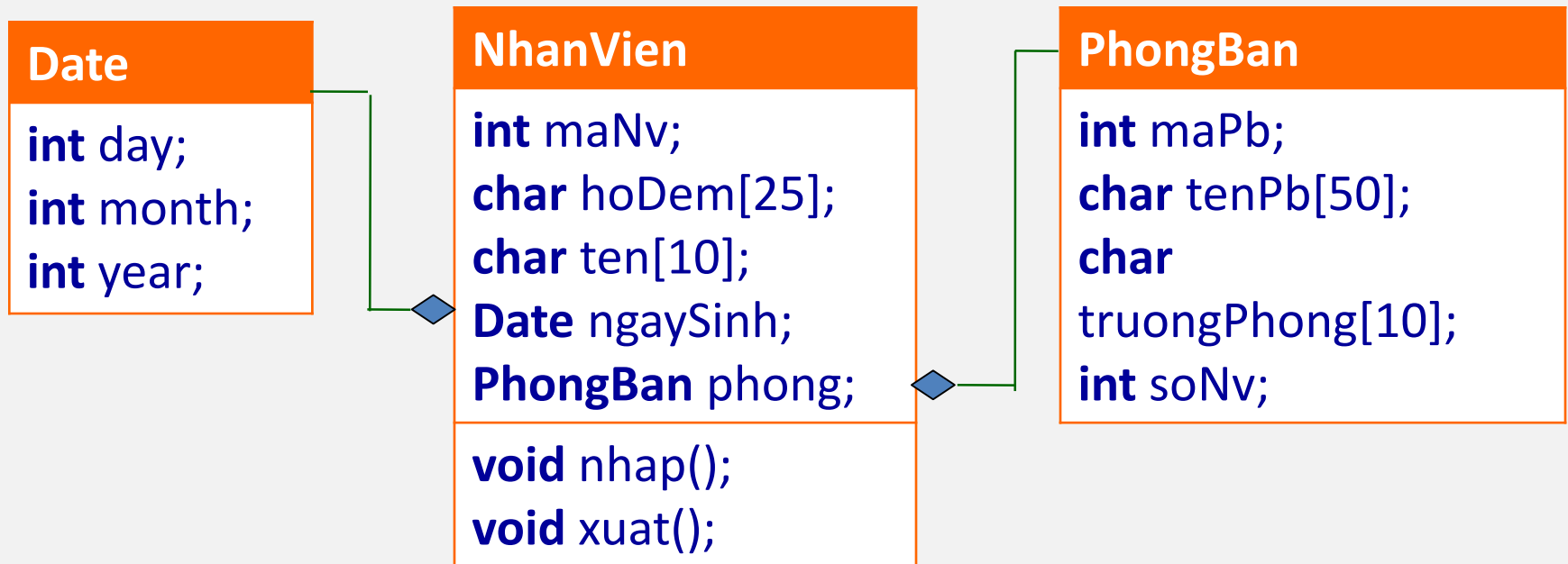


Bài tập (tt) - 03

- Cài đặt các yêu cầu chức năng:
 - Nhập danh sách n doanh nghiệp.
 - Hiển thị những doanh nghiệp ở thành phố Hà Nội ra màn hình.
 - Tính tổng doanh thu của những doanh nghiệp thành lập năm 2015.
 - Nhập vào mã của một doanh nghiệp, cho phép sửa lại toàn bộ thông tin của doanh nghiệp có mã vừa nhập (nếu có).

Bài tập (tt) - 04

- Cài đặt chương trình theo sơ đồ lớp sau:



Bài tập (tt) - 04

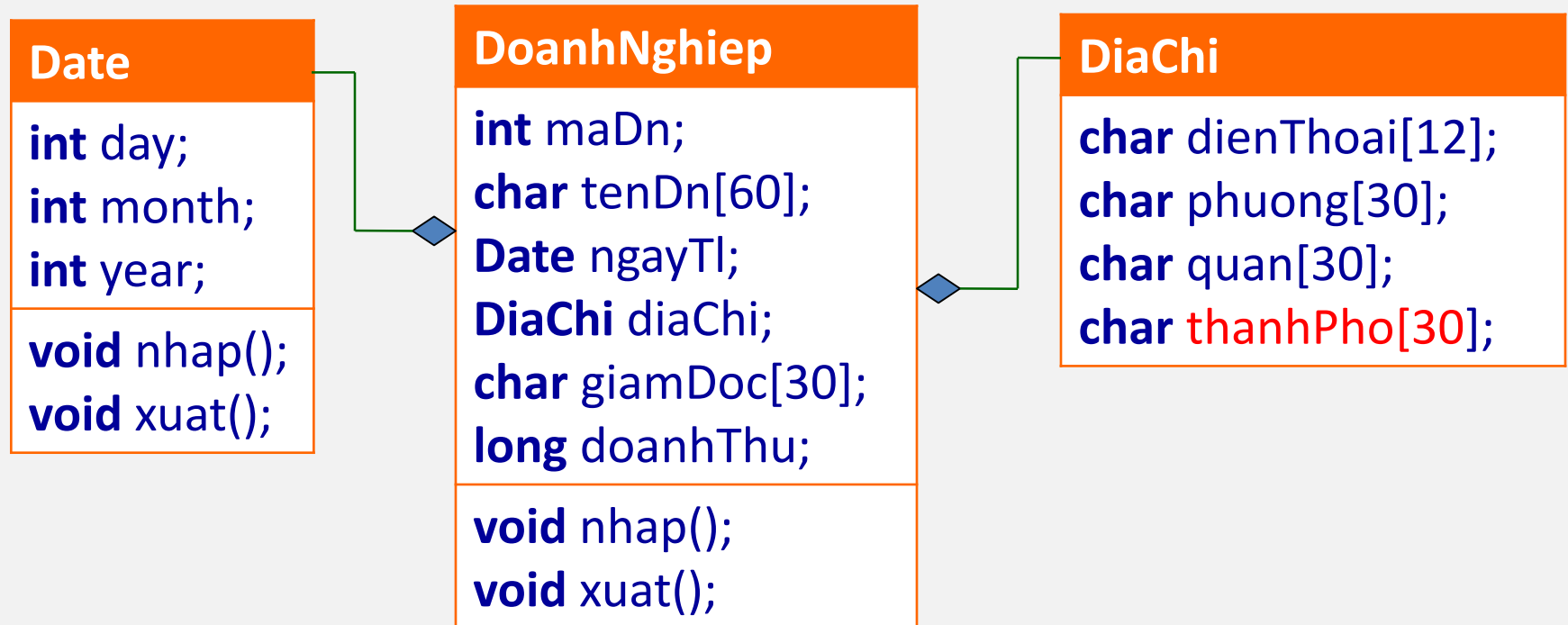
- **Cài đặt các yêu cầu chức năng:**
 - Nhập danh sách n nhân viên.
 - Hiển thị những nhân viên phòng tài chính ra màn hình.
 - Sắp xếp danh sách theo chiều tăng dần của tên nhân viên, hiển thị danh sách ra màn hình.
 - Nhập một nhân viên mới và số nguyên dương k, chèn nhân viên mới vào vị trí k trong danh sách.
 - Xóa nhân viên có mã 123.

Bài tập (tt) - 02

- Cài đặt chương trình thực hiện các yêu cầu
 - Cài đặt lớp XeHoi (xe hơi) gồm các thuộc tính: Nhãn hiệu, **hãng sản xuất**, kiểu dáng, màu sơn, năm sản xuất, xuất xứ, giá bán và các phương thức cần thiết.
 - Nhập vào một danh sách n xe hơi.
 - Hiển thị danh sách ra màn hình.
 - Hiển thị ra màn hình những xe hơi **của hãng “Toyota”**.
 - Sắp xếp danh sách theo chiều tăng dần của **giá bán**, in kết quả lên màn hình.

Bài tập (tt) - 03

- Cài đặt chương trình theo sơ đồ lớp sau:

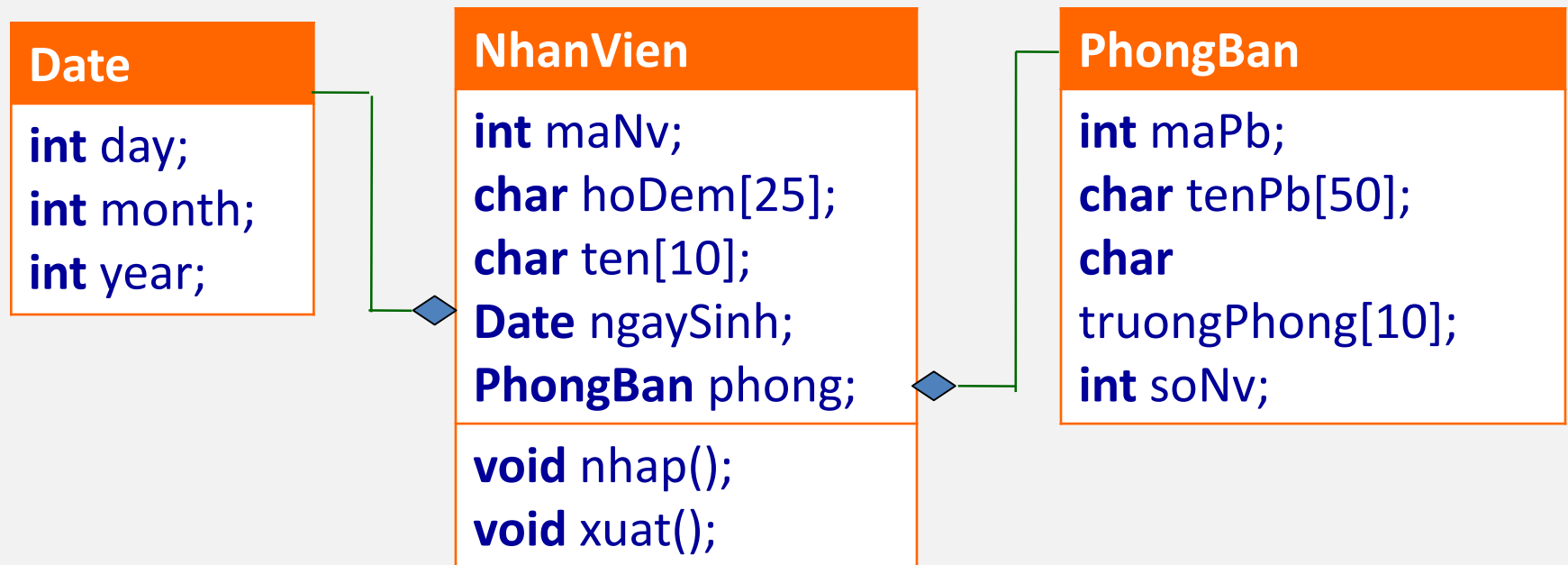


Bài tập (tt) - 03

- Cài đặt các yêu cầu chức năng:
 - Nhập danh sách n doanh nghiệp.
 - Hiển thị những doanh nghiệp ở thành phố Hà Nội ra màn hình.
 - Tính tổng doanh thu của những doanh nghiệp thành lập năm 2015.
 - Nhập vào mã của một doanh nghiệp, cho phép sửa lại toàn bộ thông tin của doanh nghiệp có mã vừa nhập (nếu có).

Bài tập (tt) - 04

- Cài đặt chương trình theo sơ đồ lớp sau:



Bài tập (tt) - 04

- Cài đặt các yêu cầu chức năng:
 - Nhập danh sách n nhân viên.
 - Hiển thị những nhân viên phòng tài chính ra màn hình.
 - Sắp xếp danh sách theo chiều tăng dần của tên nhân viên, hiển thị danh sách ra màn hình.
 - Nhập một nhân viên mới và số nguyên dương k, chèn nhân viên mới vào vị trí k trong danh sách.
 - Xóa nhân viên có mã 123.