



# BÀI TOÁN CHI TIẾT MÁY

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



# BÀI TOÁN

# Bài toán



- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
  - + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
  - + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.

# Bài toán



Yêu cầu: Thiết kế các lớp thích hợp để thực hiện các yêu cầu sau:

- Nhập các chi tiết cho máy.
- Xuất các chi tiết máy.
- Tính trị giá của máy.
- Tìm kiếm một chi tiết máy theo mã số.
- Đếm số lượng chi tiết đơn có trong cái máy (Bài tập về nhà).



# THIẾT KẾ LỚP SƠ BỘ

# Bài toán



- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
  - + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
  - + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.

# Bài toán



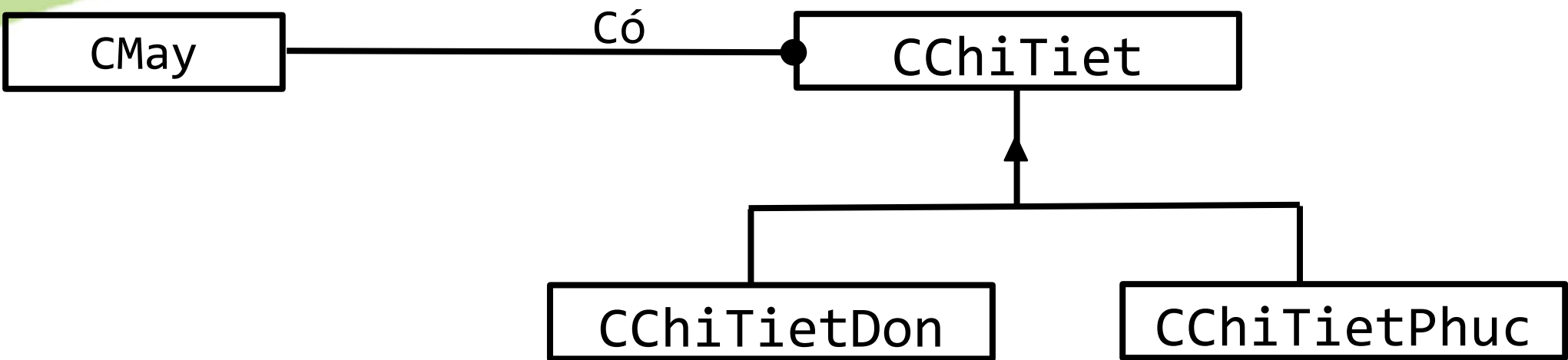
- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
- + **Chi tiết đơn**: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
- + **Chi tiết phức**: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.

# Bài toán



- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
- + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
- + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.





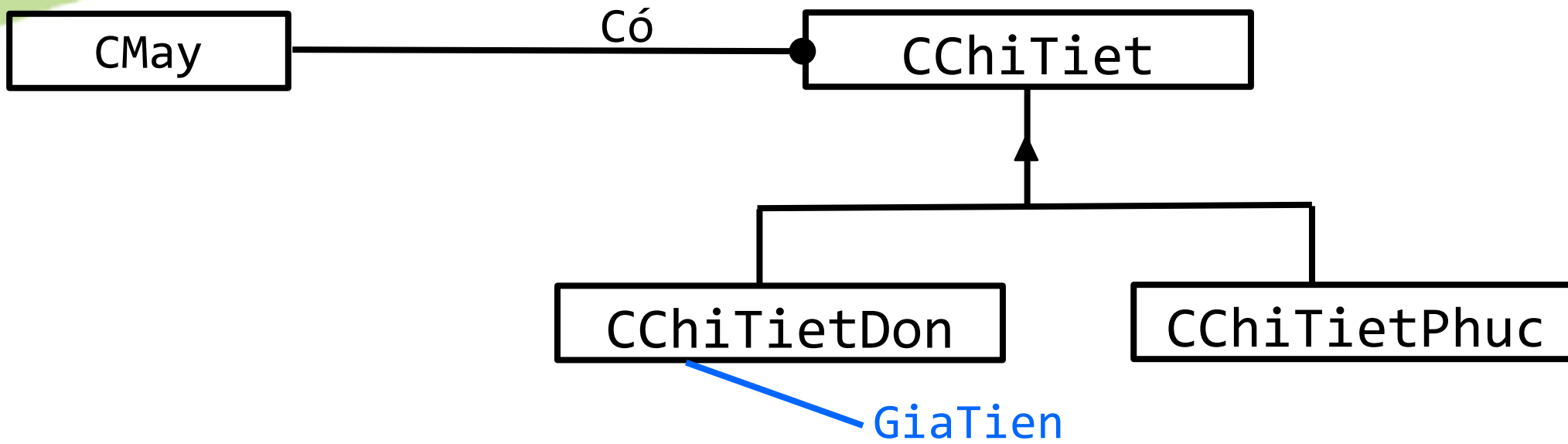


# THIẾT KẾ LỚP CHI TIẾT

# Bài toán



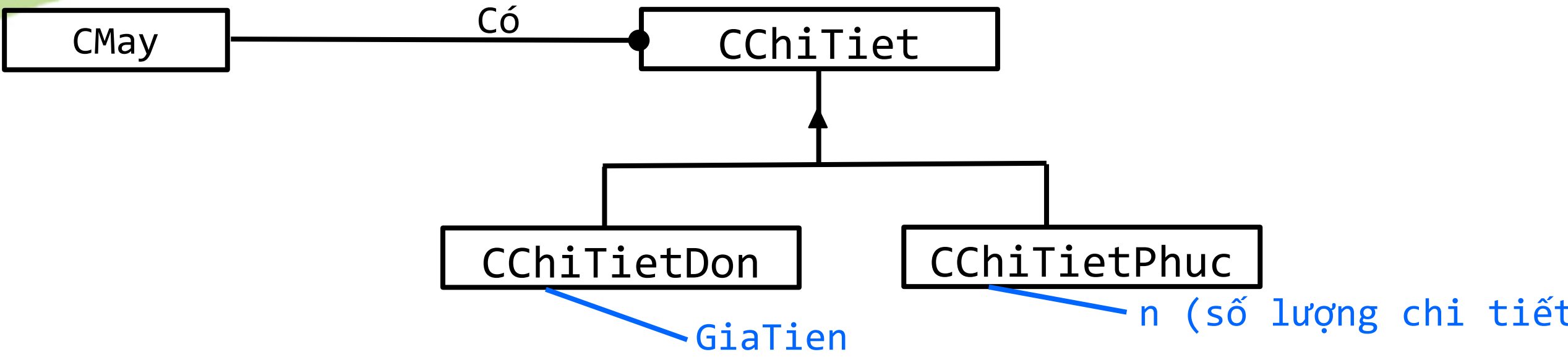
- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
  - + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
  - + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.



# Bài toán



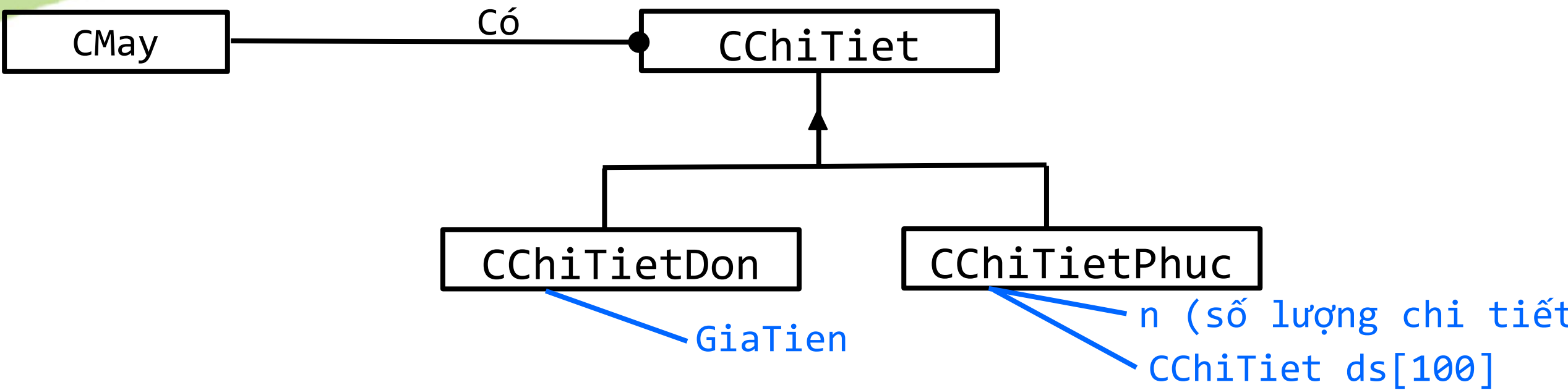
- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
  - + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
  - + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.



# Bài toán



- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
  - + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
  - + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.





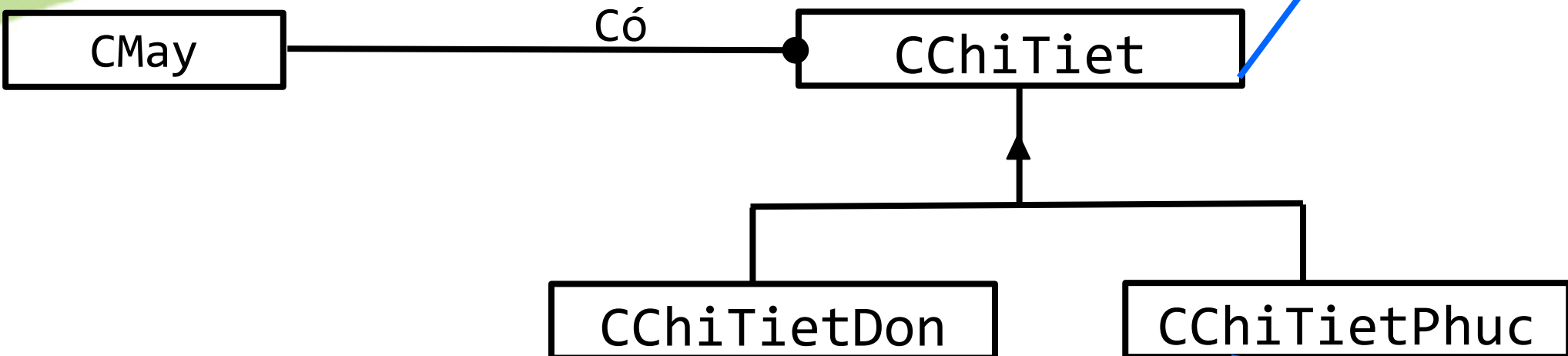
# Bài toán



- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
  - + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
  - + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.



MaSo (long)



GiaTien

n (số lượng chi tiết)  
CChiTiet ds[100]

# Bài toán



- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
  - + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
  - + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.



MaSo (long)

CMay

Có

CChiTiet

n (số lượng chi tiết)

CChiTietDon

GiaTien

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet ds[100]

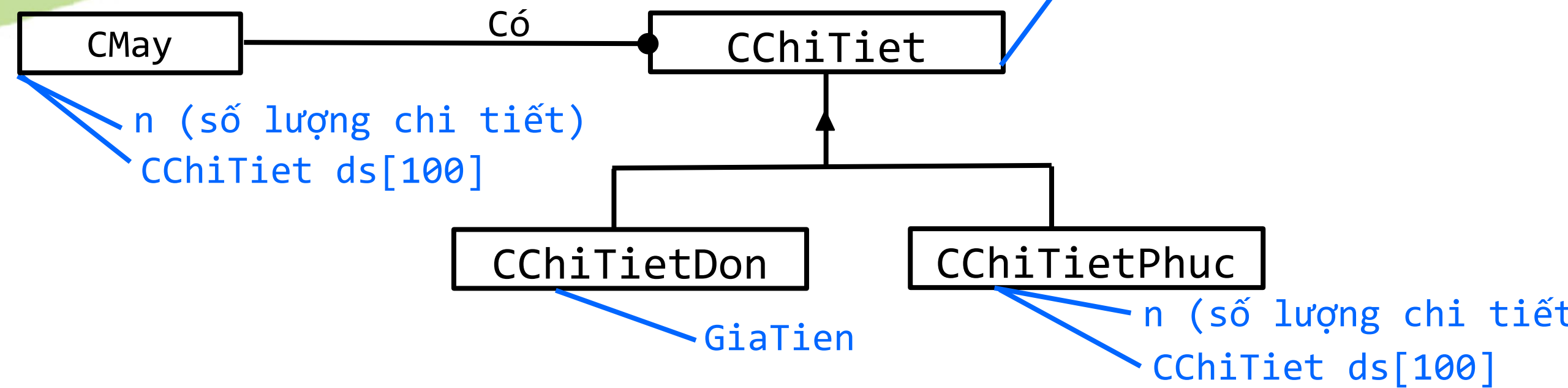
# Bài toán



- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
- + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
- + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.



MaSo (long)



# Bài toán



Yêu cầu: Thiết kế các lớp thích hợp để thực hiện các yêu cầu sau:

- Nhập các chi tiết cho máy.
- Xuất các chi tiết máy.
- Tính trị giá của máy.
- Tìm kiếm một chi tiết máy theo mã số.
- Đếm số lượng chi tiết đơn có trong cái máy (Bài tập về nhà).

# Bài toán



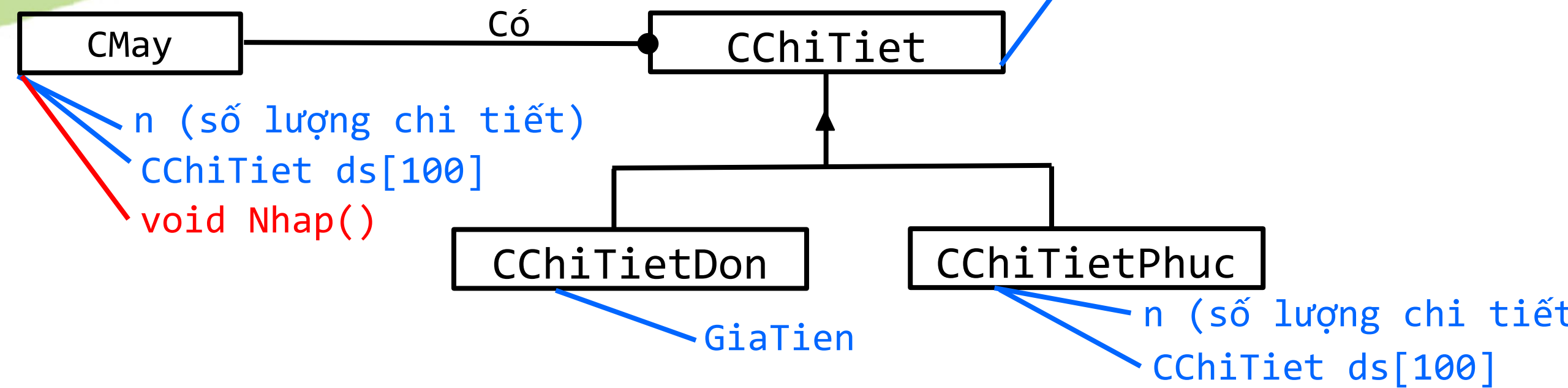
Yêu cầu: Thiết kế các lớp thích hợp để thực hiện các yêu cầu sau:

- Nhập các chi tiết cho máy.
- Xuất các chi tiết máy.
- Tính trị giá của máy.
- Tìm kiếm một chi tiết máy theo mã số.
- Đếm số lượng chi tiết đơn có trong cái máy (Bài tập về nhà).





MaSo (long)





MaSo (long)

void Nhap()

CMay

Có

CChiTiet

n (số lượng chi tiết)

CChiTiet ds[100]

void Nhap()

CChiTietDon

GiaTien

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet ds[100]



MaSo (long)

void Nhap()

Có

CMay

CChiTiet

n (số lượng chi tiết)

CChiTiet ds[100]

void Nhap()

CChiTietDon

GiaTien

void Nhap()

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet ds[100]



MaSo (long)

void Nhap()

Có

CMay

CChiTiet

n (số lượng chi tiết)

CChiTiet ds[100]

void Nhap()

CChiTietDon

GiaTien

void Nhap()

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet ds[100]

void Nhap()

# Bài toán



Yêu cầu: Thiết kế các lớp thích hợp để thực hiện các yêu cầu sau:

- Nhập các chi tiết cho máy.
- Xuất các chi tiết máy.
- Tính trị giá của máy.
- Tìm kiếm một chi tiết máy theo mã số.
- Đếm số lượng chi tiết đơn có trong cái máy (Bài tập về nhà).



MaSo (long)

void Nhap()

Có

CMay

CChiTiet

n (số lượng chi tiết)

CChiTiet ds[100]

void Nhap()

void Xuat()

CChiTietDon

GiaTien

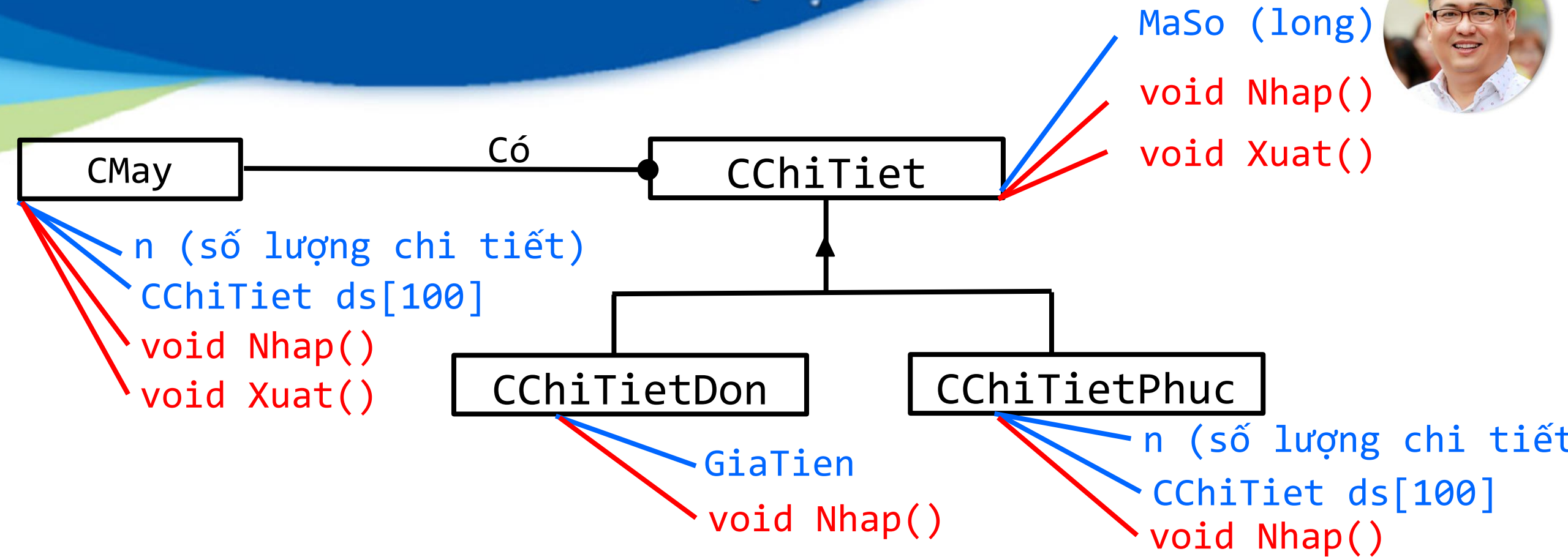
void Nhap()

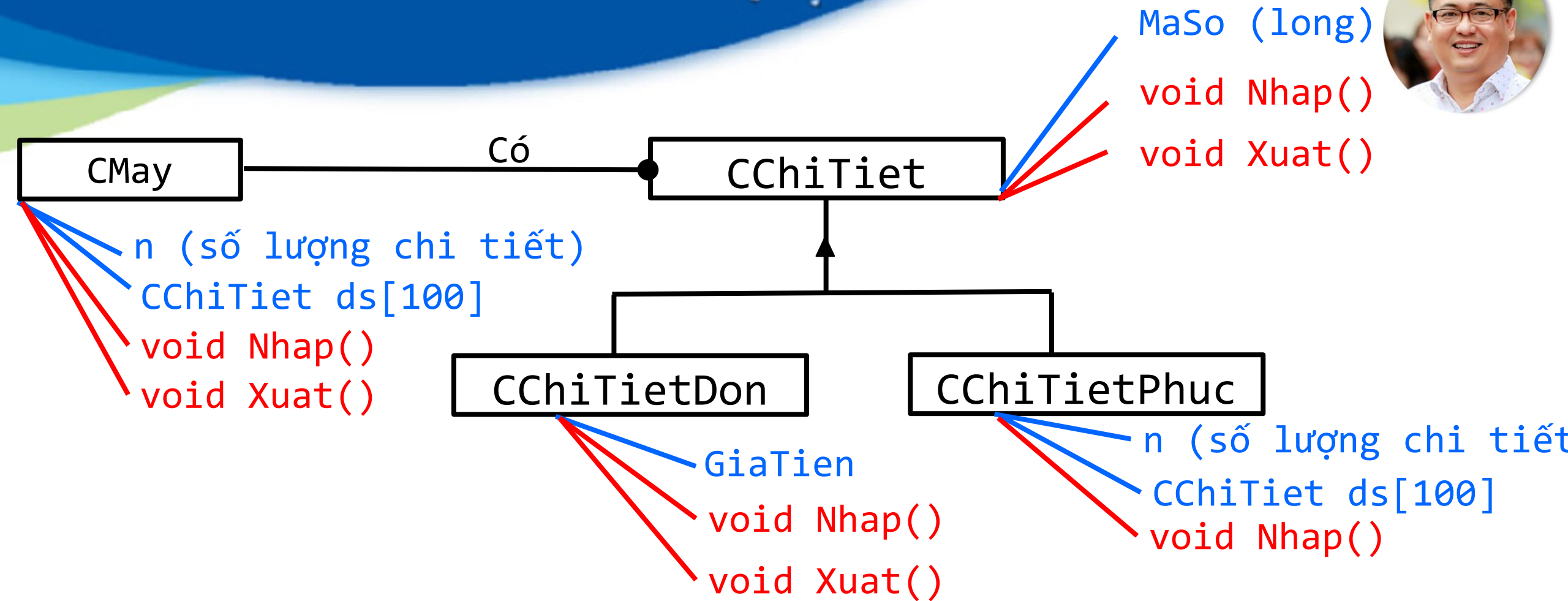
CChiTietPhuc

n (số lượng chi tiết)

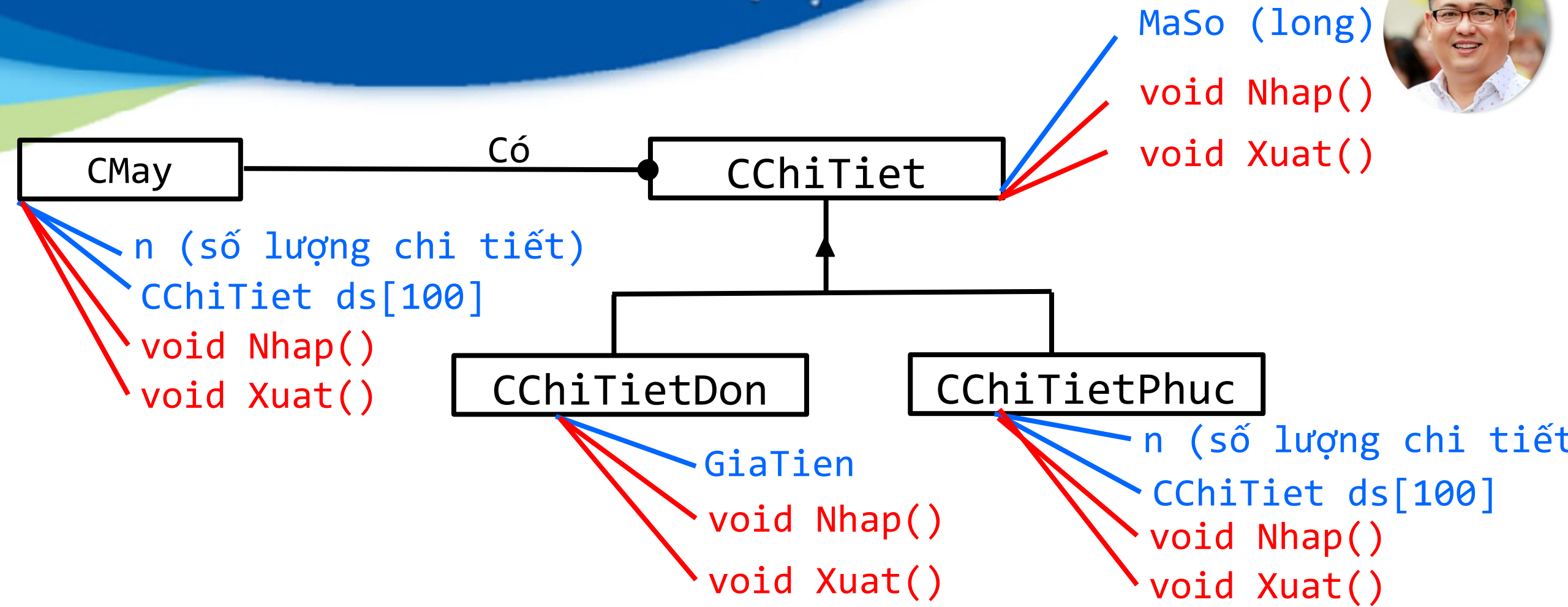
CChiTiet ds[100]

void Nhap()







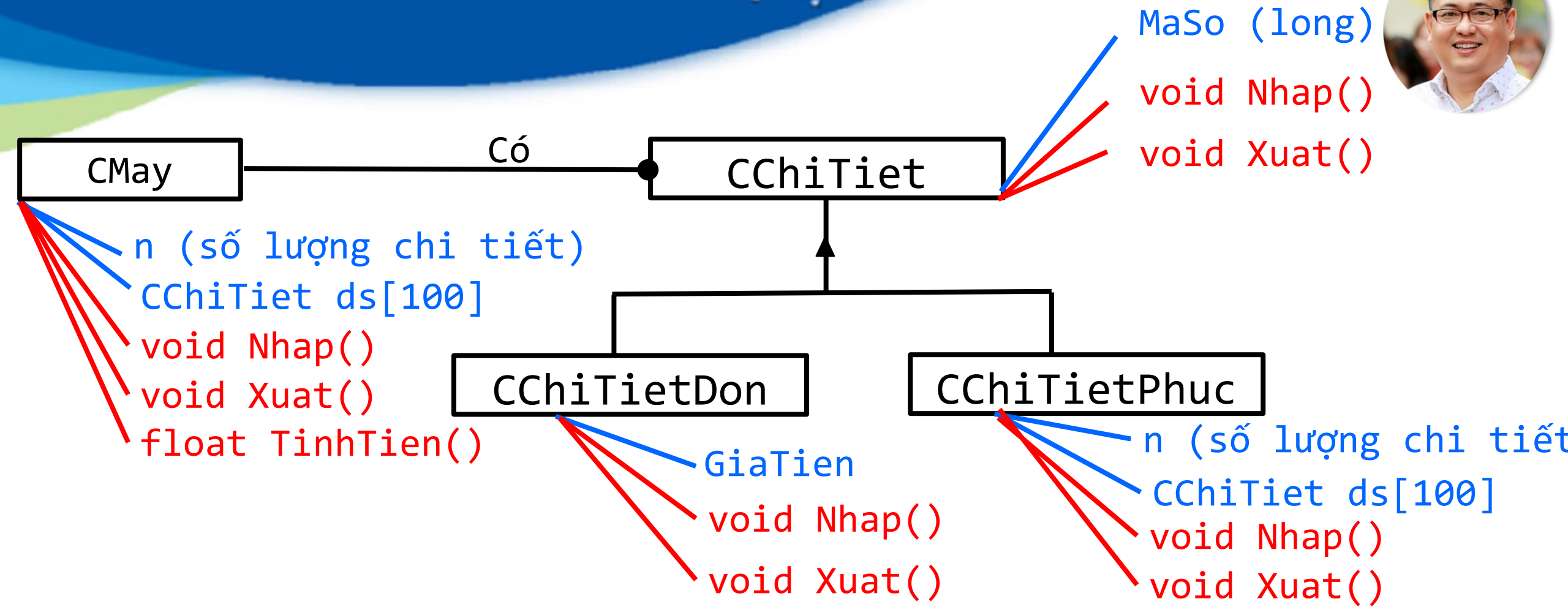


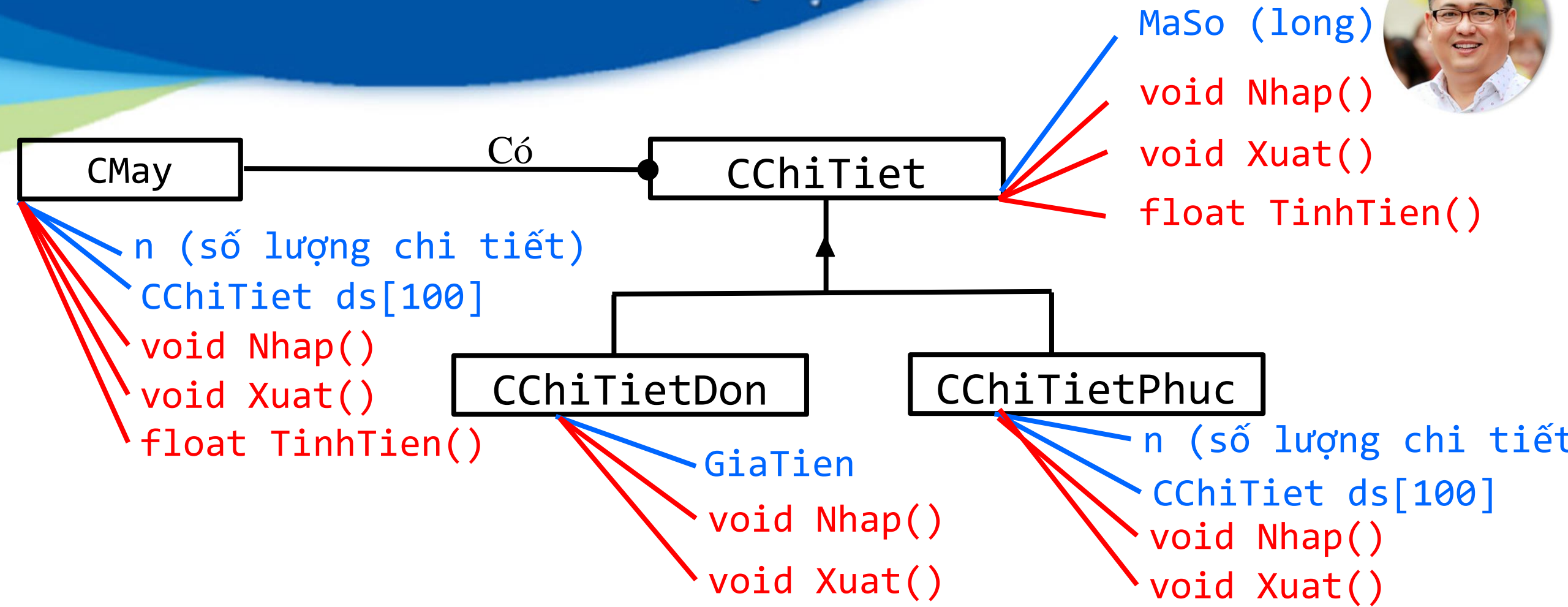
# Bài toán

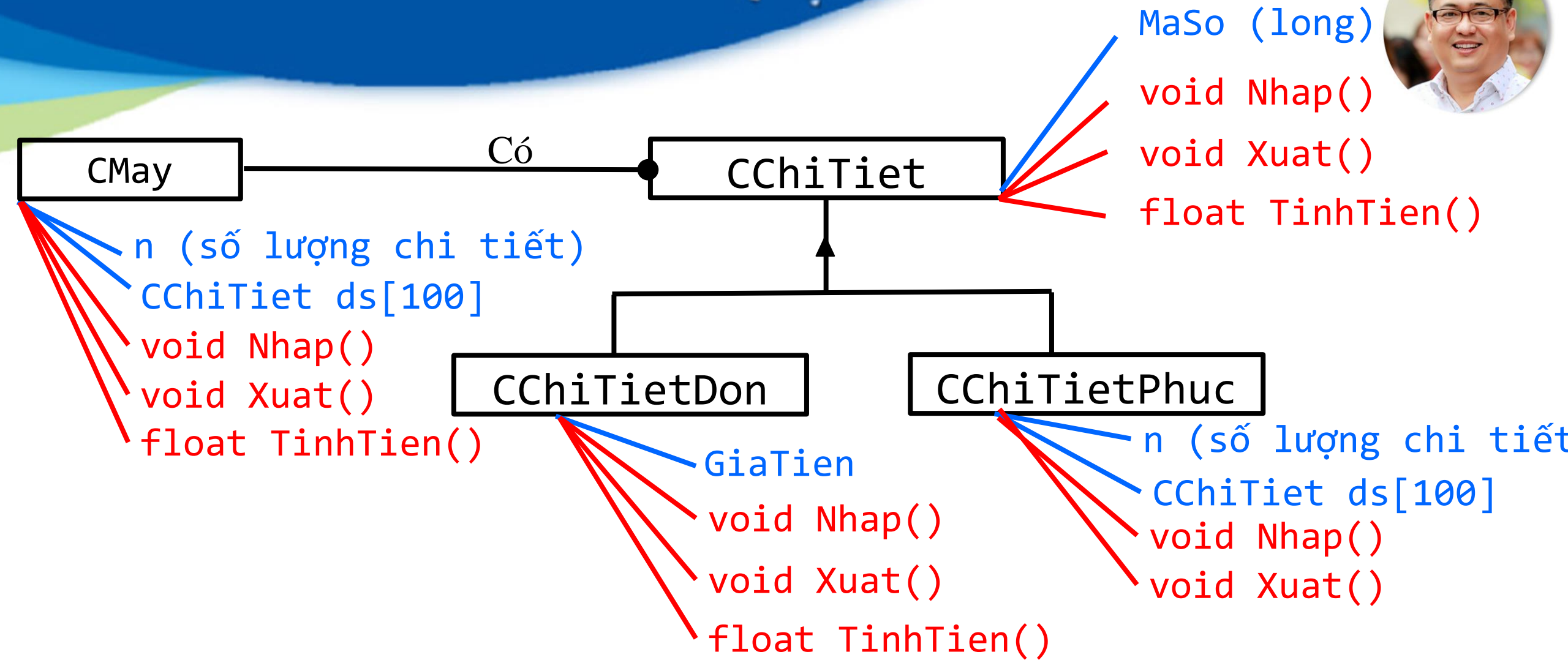


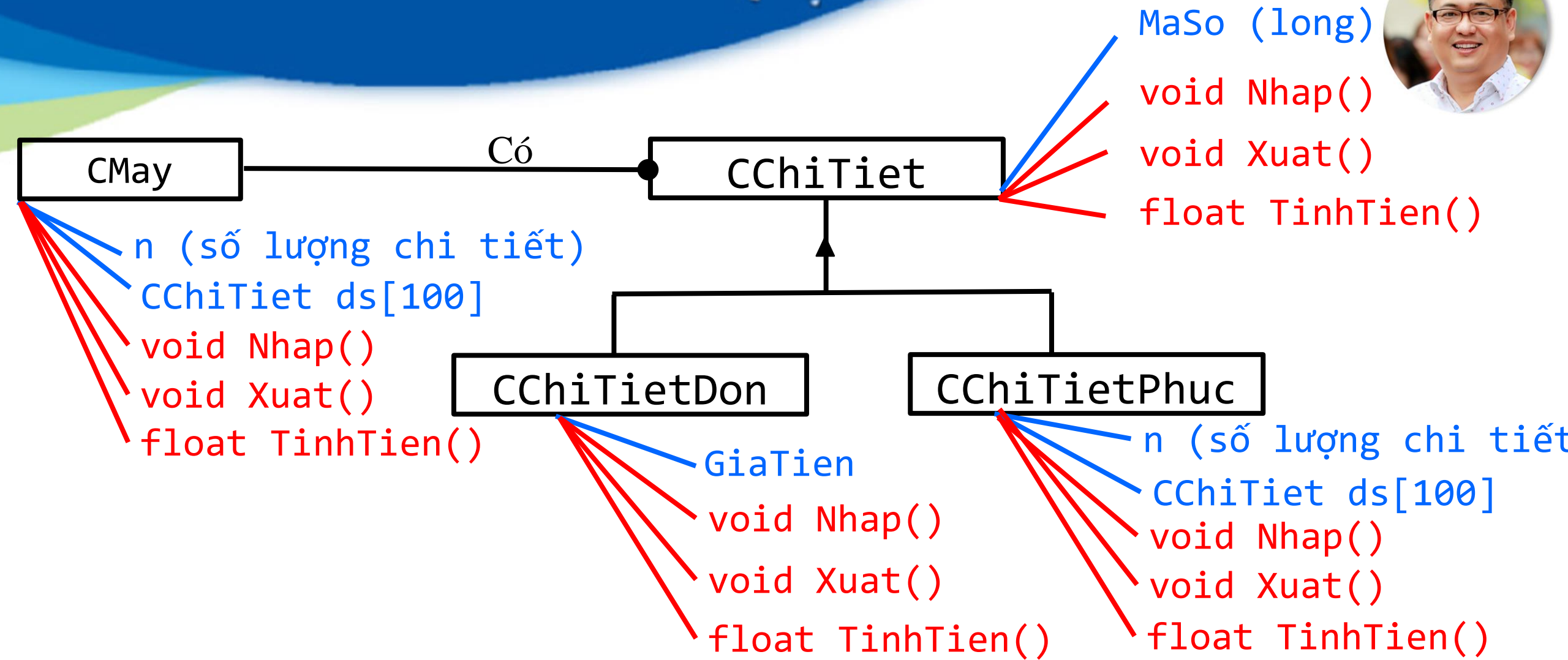
Yêu cầu: Thiết kế các lớp thích hợp để thực hiện các yêu cầu sau:

- Nhập các chi tiết cho máy.
- Xuất các chi tiết máy.
- Tính trị giá của máy.
- Tìm kiếm một chi tiết máy theo mã số.
- Đếm số lượng chi tiết đơn có trong cái máy (Bài tập về nhà).







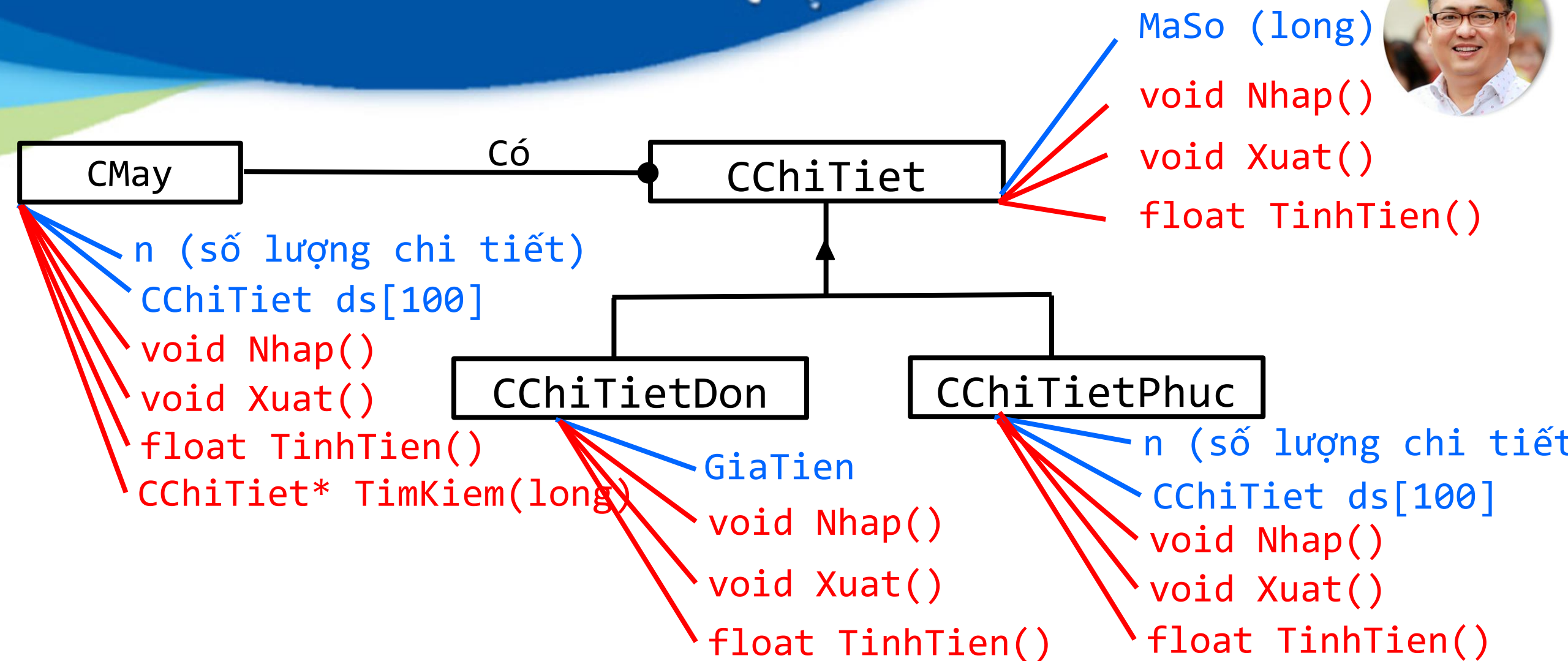


# Bài toán

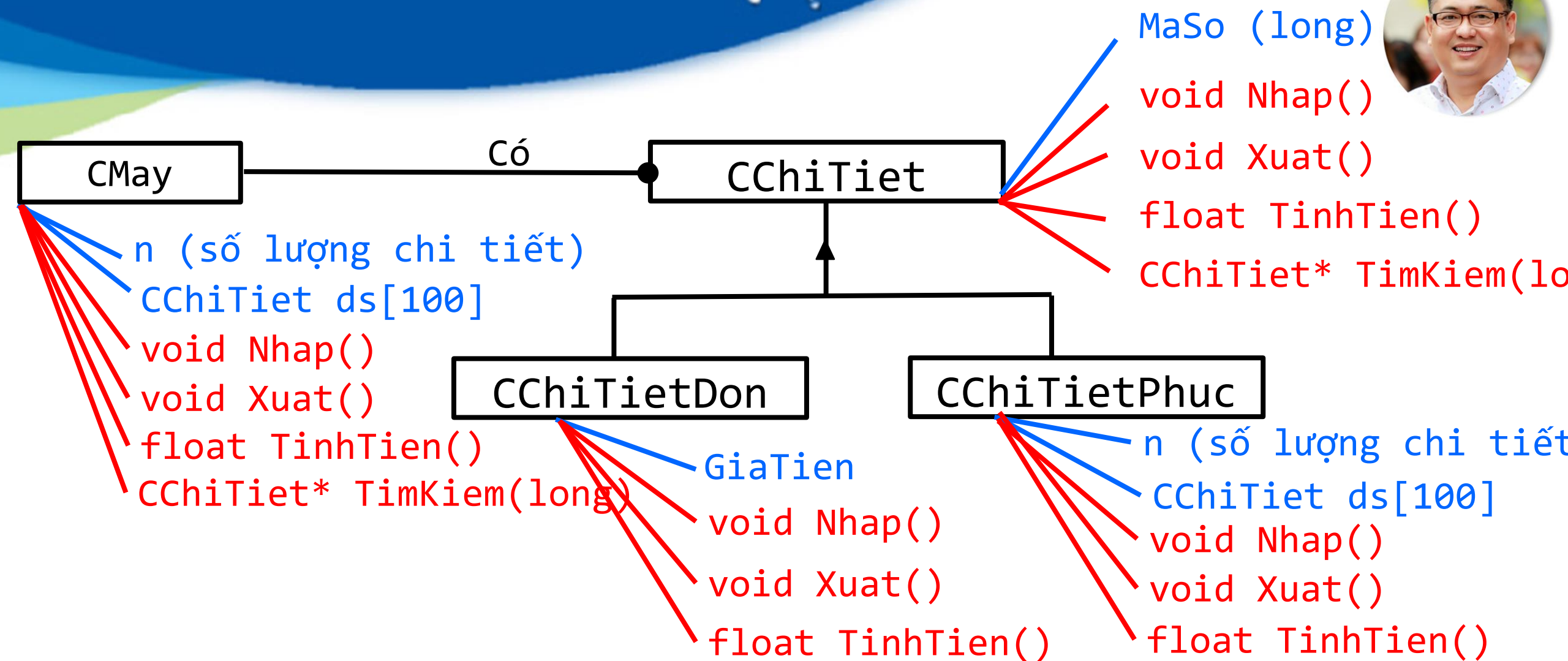


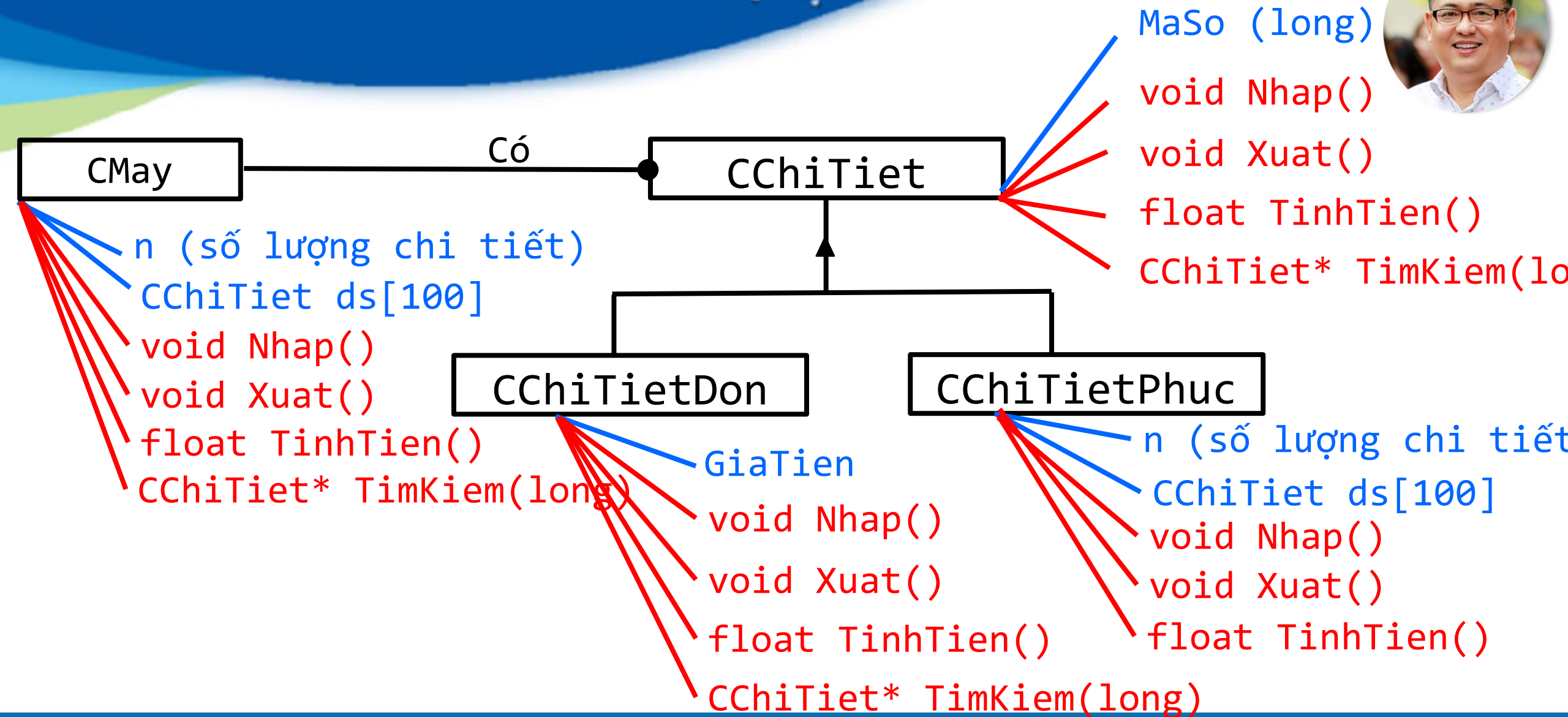
Yêu cầu: Thiết kế các lớp thích hợp để thực hiện các yêu cầu sau:

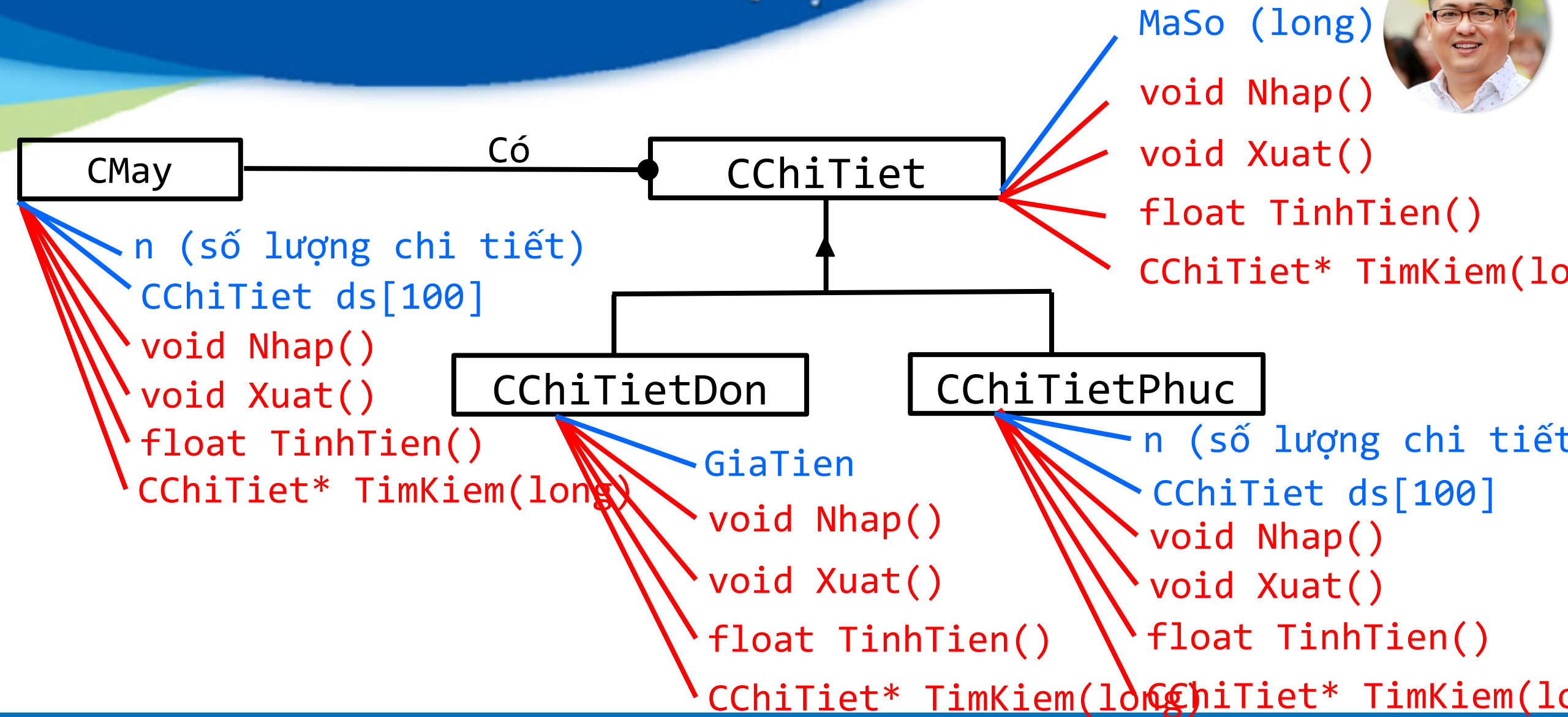
- Nhập các chi tiết cho máy.
- Xuất các chi tiết máy.
- Tính trị giá của máy.
- Tìm kiếm một chi tiết máy theo mã số.
- Đếm số lượng chi tiết đơn có trong cái máy (Bài tập về nhà).











# Bài toán



Yêu cầu: Thiết kế các lớp thích hợp để thực hiện các yêu cầu sau:

- Nhập các chi tiết cho máy.
- Xuất các chi tiết máy.
- Tính trị giá của máy.
- Tìm kiếm một chi tiết máy theo mã số.
- Đếm số lượng chi tiết đơn có trong cái máy (Bài tập về nhà).



# KHAI BÁO LỚP

# Khai báo lớp



```
11.class CChiTiet
12.{
13.    protected:
14.        long MaSo;
15.    public:
16.        void Nhap();
17.        void Xuat();
18.        float TinhTien();
19.        CChiTiet* TimKiem(long);
20.};
```

CChiTiet

MaSo (long)

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Khai báo lớp



```
11.class CChiTietDon:public CChiTiet
12.{
13.    protected:
14.        float GiaTien;
15.    public:
16.        void Nhap();
17.        void Xuat();
18.        float TinhTien();
19.        CChiTiet* TimKiem(long);
20.};
```

CChiTietDon

GiaTien

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Khai báo lớp



```
11.class CChiTietPhuc:public CChiTiet
12.{
13.    protected:
14.        int n;
15.        CChiTiet ds[100];
16.    public:
17.        void Nhap();
18.        void Xuat();
19.        float TinhTien();
20.        CChiTiet* TimKiem(long);
21.};
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

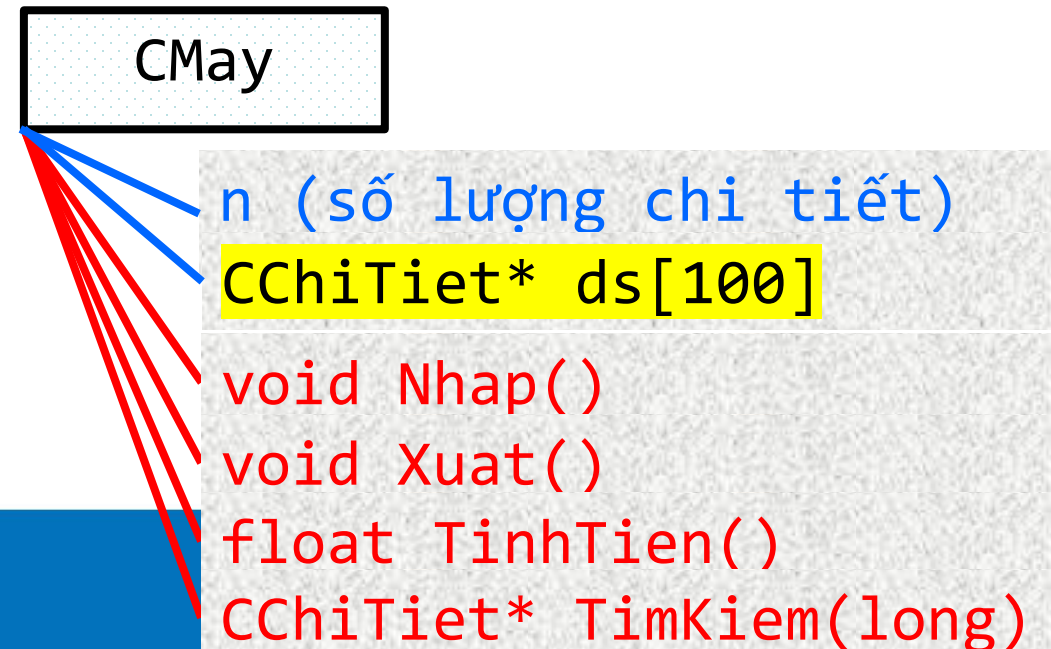
CChiTiet\* TimKiem(long)



# Khai báo lớp



```
11.class CMay
12.{
13.    protected:
14.        int n;
15.        CChiTiet* ds[100];
16.    public:
17.        void Nhap();
18.        void Xuat();
19.        float TinhTien();
20.        CChiTiet* TimKiem(long);
21.};
```





# ĐỊNH NGHĨA PHƯƠNG THỨC

# Định nghĩa các phương thức



```
11. void CChiTiet::Nhap()  
12. {  
13. |     return;  
14. }  
15. void CChiTiet::Xuat()  
16. {  
17. |     return;  
18. }
```

CChiTiet

MaSo (long)

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11.float CChiTiet::TinhTien()  
12.{  
13.|    return 0;  
14.}  
15.CChiTiet* CChiTiet::TimKiem(long ms)  
16.{  
17.|    if(MaSo==ms)  
18.|        return this;  
19.|    return NULL;  
20.}
```

CChiTiet

MaSo (long)

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11. void CChiTietDon::Nhap()  
12. {  
13.     cout << "Nhap ma so:";  
14.     cin >> MaSo;  
15.     cout << "Nhap gia:";  
16.     cin >> GiaTien;  
17. }
```

```
11. void CChiTietDon::Xuat()  
12. {  
13.     cout << "Ma so:" << MaSo;  
14.     cout << "Gia:" << GiaTien  
15. }
```

CChiTietDon

GiaTien

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11.float CChiTietDon::TinhTien()  
12.{  
13.|    return GiaTien;  
14.}  
15.CChiTiet* CChiTietDon::TimKiem(long ms)  
16.{  
17.|    if(MaSo==ms)  
18.|        return this;  
19.|    return NULL;  
20.}
```

CChiTietDon

GiaTien

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



- Bài toán: Một cái máy có nhiều chi tiết. Mỗi chi tiết máy có thể là chi tiết đơn hoặc là chi tiết phức.
  - + Chi tiết đơn: là chi tiết không chứa bên trong nó chi tiết khác. Thông tin của chi tiết đơn bao gồm: mã số chi tiết, giá tiền.
  - + Chi tiết phức: là chi tiết chứa bên trong nó nhiều chi tiết thành phần, mỗi một chi tiết thành phần này có thể là chi tiết đơn hoặc là chi tiết phức. Thông tin của chi tiết phức bao gồm: mã số chi tiết, số lượng chi tiết thành phần, danh sách các chi tiết thành phần. Giá tiền của chi tiết phức bằng tổng giá tiền của các chi tiết thành phần.



# Định nghĩa các phương thức



```
11.float CChiTietPhuc::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i].TinhTien();  
16.    return s;  
17.}
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

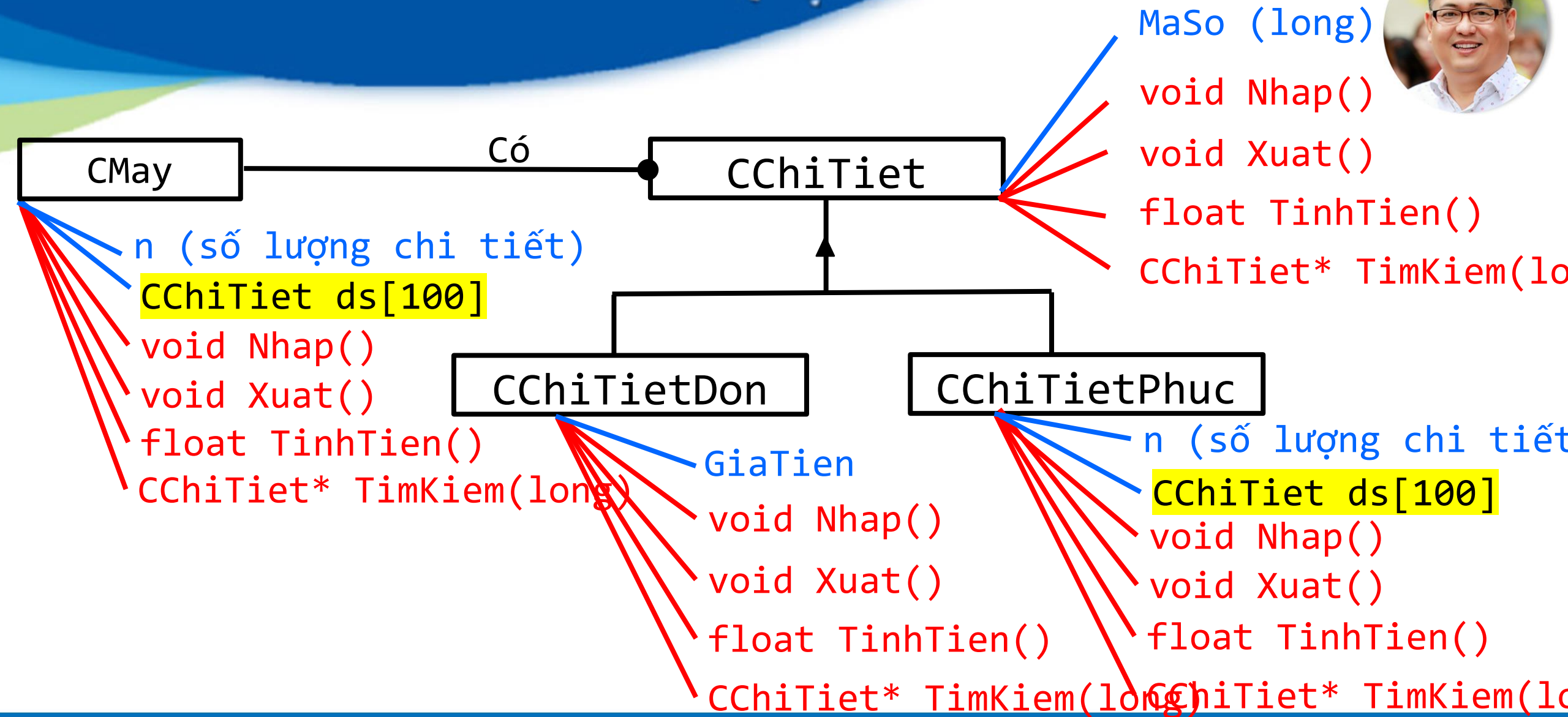
voidNhap()

voidXuat()

floatTinhTien()

CChiTiet\* TimKiem(long)





# Định nghĩa các phương thức



```
11.float CChiTiet::TinhTien()  
12.{  
13.|    return 0;  
14.}  
15.CChiTiet* CChiTiet::TimKiem(long ms)  
16.{  
17.|    if(MaSo==ms)  
18.|        return this;  
19.|    return NULL;  
20.}
```

CChiTiet

MaSo (long)

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11.float CChiTietPhuc::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i].TinhTien(): SAI  
16.    return s;  
17.}
```

CChiTietPhuc

n (số lượng chi tiết)

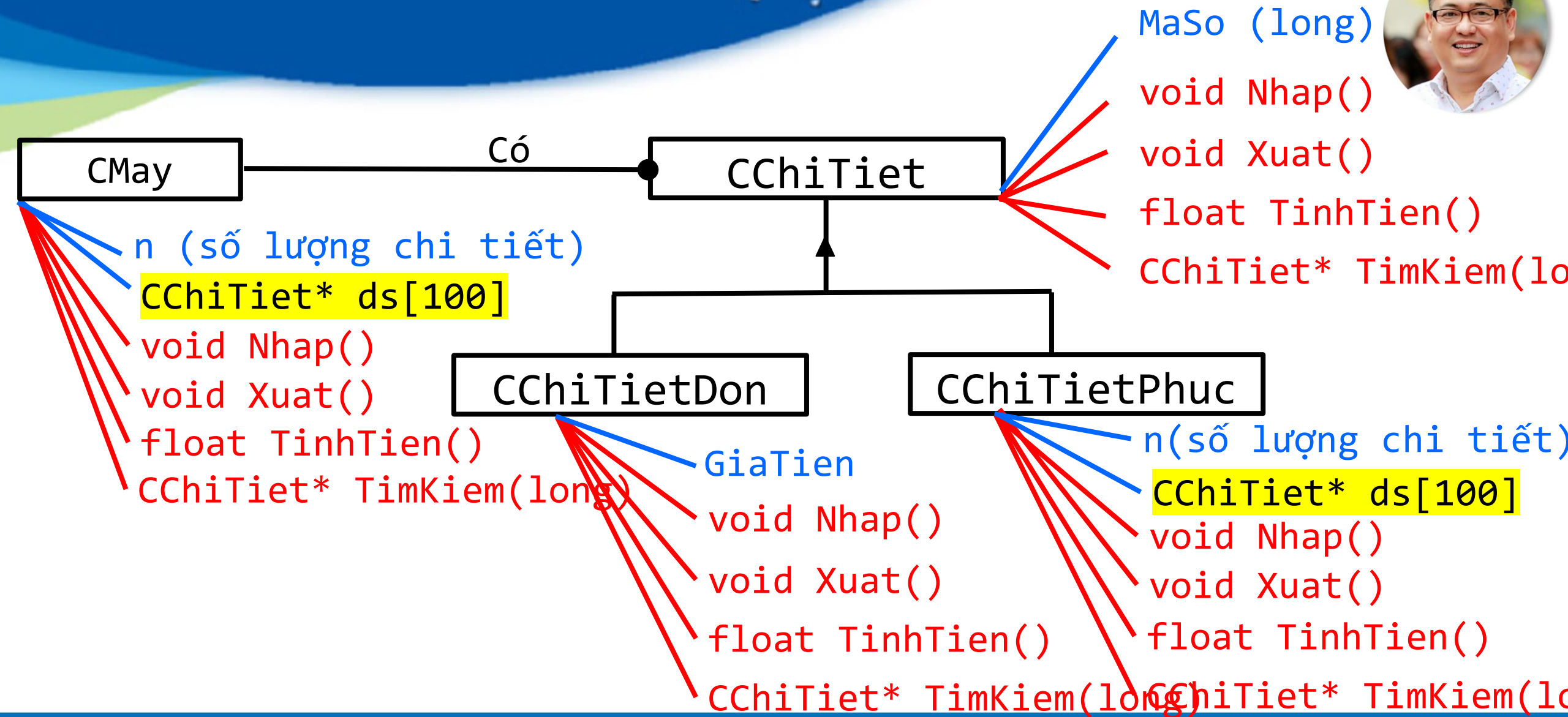
CChiTiet\* ds[100]

voidNhap()

voidXuat()

floatTinhTien()

CChiTiet\* TimKiem(long)



# Khai báo lớp



```
11.class CChiTietPhuc:public CChiTiet
12.{
13.    protected:
14.        int n;
15.        CChiTiet* ds[100];
16.    public:
17.        void Nhap();
18.        void Xuat();
19.        float TinhTien();
20.        CChiTiet* TimKiem(long);
21.};
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Khai báo lớp



```
11.class CMay
12.{
13.    protected:
14.        int n;
15.        CChiTiet* ds[100];
16.    public:
17.        void Nhap();
18.        void Xuat();
19.        float TinhTien();
20.        CChiTiet* TimKiem(long);
21.};
```

CMay

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)



# Định nghĩa các phương thức



```
11.float CChiTietPhuc::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i]->TinhTien();  
16.    return s;  
17.}
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

voidNhap()

voidXuat()

floatTinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11.float CChiTiet::TinhTien()  
12.{  
13.|    return 0;  
14.}  
15.CChiTiet* CChiTiet::TimKiem(long ms)  
16.{  
17.|    if(MaSo==ms)  
18.|        return this;  
19.|    return NULL;  
20.}
```

CChiTiet

MaSo (long)

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)



# Định nghĩa các phương thức



```
11.float CChiTietPhuc::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i]->TinhTien();  
16.    return s;  
17.}
```

SAI LẦM HẠI

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

voidNhap()

voidXuat()

floatTinhTien()

CChiTiet\* TimKiem(long)

# Nhắc lại lý thuyết đa xạ



- Khái niệm: Đa xạ là cơ chế **tầm vực động** (**dynamic scope**) cho phép "**xác định**" đúng hành vi (phương thức – **method**) của đối tượng (**object**) khi yêu cầu thực hiện.
- Việc "**xác định**" được thực hiện theo nguyên tắc tự nhiên: con trỏ đối tượng đang giữ địa chỉ của đối tượng thuộc về lớp nào thì nó sẽ gọi thực hiện phương thức của lớp đối tượng (**class**) đó.
- Tầm vực động (**dynamic scope**) là cơ chế gọi thực hiện phương thức thông qua con trỏ đối tượng.

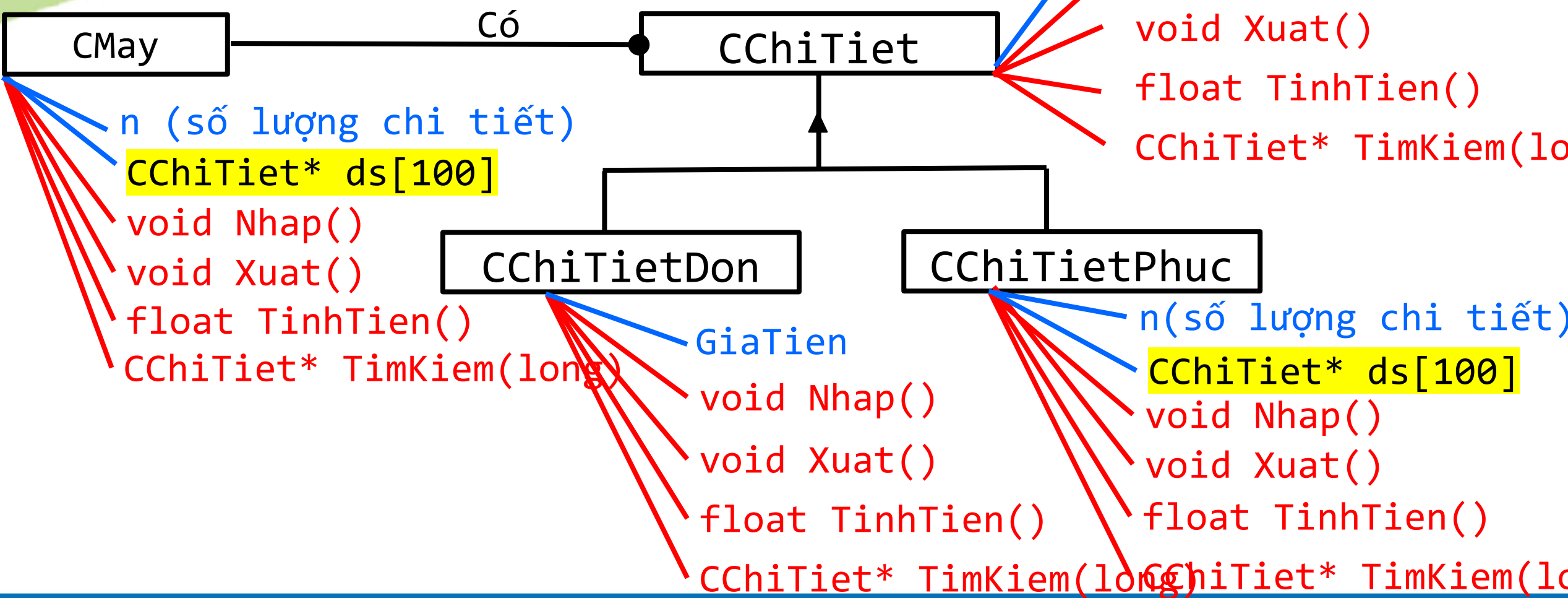
# Nhắc lại lý thuyết đa xạ



- Một phương thức được khai báo bắt đầu với từ khóa **virtual** thì được gọi là phương thức ảo và phương thức này được gọi thực hiện theo cơ chế đa xạ nếu **lời gọi thực hiện phương thức** được thông qua một con trỏ đối tượng.
- Các phương thức ở lớp dẫn xuất cùng tên và cùng danh sách tham số đầu vào thì cũng sẽ là phương thức ảo nếu ở lớp cơ sở phương thức cùng tên và cùng tham số là phương thức ảo.



virtual



# Khai báo lớp



```
11.class CChiTiet
12.{
13.    protected:
14.        long MaSo;
15.    public:
16.        virtual void Nhap();
17.        virtual void Xuat();
18.        virtual float TinhTien();
19.        virtual CChiTiet* TimKiem(long);
20.};
```



# Khai báo lớp

```
11.class CChiTietPhuc:public CChiTiet
12.{
13.    protected:
14.        int n;
15.        CChiTiet* ds[100];
16.    public:
17.        void Nhap();
18.        void Xuat();
19.        float TinhTien();
20.        CChiTiet* TimKiem(long);
21.};
```

**virtual**

Các phương thức ở lớp dẫn xuất cùng tên và cùng danh sách tham số đầu vào thì cũng sẽ là phương thức ảo nếu ở lớp cơ sở phương thức cùng tên và cùng tham số là phương thức

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)



# Khai báo lớp

```
11.class CChiTietDon:public CChiTiet
12.{
13.    protected:
14.        float GiaTien;
15.    public:
16.        void Nhap();
17.        void Xuat();
18.        float TinhTien();
19.        CChiTiet* TimKiem(long);
20.};
```

virtual

Các phương thức ở lớp dẫn xuất cùng tên và cùng danh sách tham số đầu vào thì cũng sẽ là phương thức ảo nếu ở lớp cơ sở phương thức cùng tên và cùng tham số là phương thức ảo.

CChiTietDon

GiaTien

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)



# Định nghĩa các phương thức



```
11.float CChiTietPhuc::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i]->TinhTien();  
16.    return s;  
17.}
```

ĐÚNG

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)



# Định nghĩa các phương thức



```
11.float CChiTietPhuc::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i]->TinhTien(); ĐÚNG  
16.    return s;  
17.}
```

- Dòng 15 ta nói con trỏ đối tượng `ds[i]` gọi thực thực hiện phương thức `TinhTien` và đây là phương thức ảo nên việc gọi thực hiện phương thức này sẽ theo cơ chế đa xạ.

# Định nghĩa các phương thức



```
11.float CChiTietPhuc::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i]->TinhTien();  
16.    return s;  
17.}
```

Nếu con trỏ đối tượng `ds[i]` giữ địa chỉ của đối tượng thuộc về lớp `CChiTietDon` thì chương trình sẽ gọi thực hiện phương thức tính tiền của lớp `CChiTietDon`.

— Dòng 15 ta nói con trỏ đối tượng `ds[i]` gọi thực thực hiện phương thức `TinhTien` và đây là phương thức ảo nên việc gọi thực hiện phương thức này sẽ theo cơ chế đa xạ.

# Định nghĩa các phương thức



```
11.float CChiTietPhuc::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i]->TinhTien();  
16.    return s;  
17.}
```

Nếu con trỏ đối tượng `ds[i]` giữ địa chỉ của đối tượng thuộc về lớp `CChiTietPhuc` thì chương trình sẽ gọi thực hiện phương thức tính tiền của lớp `CChiTietPhuc`.

- Dòng 15 ta nói con trỏ đối tượng `ds[i]` gọi thực thực hiện phương thức `TinhTien` và đây là phương thức ảo nên việc gọi thực hiện phương thức này sẽ theo cơ chế đa xạ.

# Nhắc lại lý thuyết đa xạ



- Khái niệm: Đa xạ là cơ chế **tầm vực động** (dynamic scope) cho phép "**xác định**" đúng hành vi (phương thức – **method**) của đối tượng (**object**) khi yêu cầu thực hiện.
- Việc "**xác định**" được thực hiện theo nguyên tắc tự nhiên: con trỏ đối tượng đang giữ địa chỉ của đối tượng thuộc về lớp nào thì nó sẽ gọi thực hiện phương thức của lớp đối tượng (**class**) đó.
- Tầm vực động (dynamic scope) là cơ chế gọi thực hiện phương thức thông qua con trỏ đối tượng.

# Định nghĩa các phương thức



```
11. void CChiTietPhuc::Xuat()  
12. {  
13.     cout << "\n Ma so:" << MaSo;  
14.     cout << "\n So lượng chi tiet thanh phan:" << n;  
15.     for(int i=0; i<n; i++)  
16.         ds[i]->Xuat();  
17. }
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

voidNhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11.CChiTiet* CChiTietPhuc::TimKiem(long ms)
12.{
13.    if(MaSo==ms)
14.        return this;
15.    for(int i=0; i<n; i++)
16.    {
17.        CChiTiet* kq = ds[i]->TimKiem(ms);
18.        if(kq!=NULL)
19.            return kq;
20.    }
21.    return NULL;
22.}
```

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11. void CChiTietPhuc::Nhap()  
12. {  
13.     cout << "Nhap ma so:";  
14.     cin >> MaSo;  
15.     cout << "Nhap so luong chi tiet thanh phan:";  
16.     cin >> n;  
17.     for(int i=0; i<n; i++)  
18.     {  
19.         cout << "Nhap chi tiet a[" << i << "]:";  
20.         ds[i]->Nhap();  
21.     }  
22. }
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)



# Định nghĩa các phương thức



```
11. void CChiTietPhuc::Nhap()  
12. {  
13.     cout << "Nhap ma so:";  
14.     cin >> MaSo;  
15.     cout << "Nhap so luong chi tiet thanh phan:";  
16.     cin >> n;  
17.     for(int i=0; i<n; i++)  
18.     {  
19.         cout << "Nhap chi tiet a[" << i << ": ";  
20.         ds[i]->Nhap(); SAI  
21.     }  
22. }
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)





```
11. void CChiTietPhuc::Nhap()
12. {
13.     cout << "Nhap ma so:";
14.     cin >> MaSo;
15.     cout << "Nhap so luong chi tiet thanh phan:";
16.     cin >> n;
17.     for(int i=0; i<n; i++)
18.     {
19.         int loai;
20.         cout << "Nhap loai (0. Don, 1. Phuc):";
21.         cin >> loai;
22.         switch(loai)
23.         {
24.             case 0: ds[i] = new CChiTietDon;
25.                 break;
26.             case 1: ds[i] = new CChiTietPhuc;
27.                 break;
28.         }
29.         ds[i]->Nhap();
30.     }
31. }
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)



```
11. void CMay::Nhap()  
12. {  
13.     cout << "Nhap so luong chi tiet thanh phan:";  
14.     cin >> n;  
15.     for(int i=0; i<n; i++)  
16.     {  
17.         int loai;  
18.         cout << "Nhap loai (0. Don, 1. Phuc):";  
19.         cin >> loai;  
20.         switch(loai)  
21.         {  
22.             case 0: ds[i] = new CChiTietDon;  
23.                 break;  
24.             case 1: ds[i] = new CChiTietPhuc;  
25.                 break;  
26.         }  
27.         ds[i]->Nhap();  
28.     }  
29. }
```

CChiTietPhuc

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11. void CMay::Xuat()  
12. {  
13.     cout << "\n So lượng chi tiet thanh phan:" << n;  
14.     for(int i=0; i<n; i++)  
15.         ds[i]->Xuat();  
16. }
```

CMay

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11.CChiTiet* CMay::TimKiem(long ms)
12.{
13.    for(int i=0; i<n; i++)
14.    {
15.        CChiTiet* kq = ds[i]->TimKiem(ms);
16.        if(kq!=NULL)
17.            return kq;
18.    }
19.    return NULL;
20.}
```

CMay

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)

# Định nghĩa các phương thức



```
11.float CMay::TinhTien()  
12.{  
13.    float s = 0;  
14.    for(int i=0; i<n; i++)  
15.        s = s + ds[i]->TinhTien();  
16.    return s;  
17.}
```

CMay

n (số lượng chi tiết)

CChiTiet\* ds[100]

void Nhap()

void Xuat()

float TinhTien()

CChiTiet\* TimKiem(long)



**Cảm ơn quý vị đã lắng nghe**

**Nhóm tác giả**

**Hồ Thái Ngọc**

**ThS. Võ Duy Nguyên**

**TS. Nguyễn Tấn Trần Minh Khang**