

# Chương 4

## SƠ LƯỢC THƯ VIỆN IOSTREAM

1. TS. Nguyễn Tấn Trần Minh Khang
2. ThS. Võ Duy Nguyên
3. ThS. Nguyễn Hoàng Ngân
4. Hồ Thái Ngọc – Source code.

# 1. VÍ DỤ DẪN NHẬP 1

# 1. VÍ DỤ DẪN NHẬP 1

- Bài toán: Viết lệnh nhập giá trị cho một số nguyên a và xuất số nguyên ra màn hình bằng cách sử dụng thư viện `iostream`.
- Phong cách 01: Sử dụng thư viện `stdio.h` (*standard input output*) và thư viện `conio.h` (*console input output*).

```
1. int a;
2. printf("Nhap mot so nguyen:");
3. scanf("%d", &a);
4. printf("So nguyen vua nhap:%d", a);
```

# 1. VÍ DỤ DẪN NHẬP 1

- Bài toán: Viết lệnh nhập giá trị cho một số nguyên a và xuất số nguyên ra màn hình bằng cách sử dụng thư viện `iostream`.
- Phong cách 02: Sử dụng thư viện `iostream`.
  1. `int a;`
  2. `cout<<"Nhập một số nguyên:";`
  3. `cin>>a;`
  4. `cout<<"Số nguyên vừa nhập:"<<a;`
- Nhắc lại:
  - + Ký hiệu `>>` được gọi là **toán tử vào, toán tử nhập**.
  - + Ký hiệu `<<` được gọi là **toán tử ra, toán tử xuất**.

## 2. VÍ DỤ DẪN NHẬP 2

## 2. VÍ DỤ DẪN NHẬP 2

- Bài toán: Viết hàm nhập thông tin của một phân số bằng cách sử dụng thư viện `iostream`.
- Cấu trúc dữ liệu:
  1. `struct phanso`
  2. `{`
  3. `int tu;`
  4. `int mau;`
  5. `};`
  6. `typedef struct phanso PHANSO;`
- Định nghĩa hàm:

## 2. VÍ DỤ DẪN NHẬP 2

```

11. void Nhap (PHANSO &x)
12. {
13.     cout<<"Nhap tu:";
14.     cin>>x.tu;
15.     cout<<"Nhap mau:";
16.     cin>>x.mau;
17. }
18. void Xuat (PHANSO x)
19. {
20.     cout<<x.tu<<"/"<<x.mau;
21. }
    
```

# 3. ĐẶT VẤN ĐỀ



# 3. ĐẶT VẤN ĐỀ

## — Nhập xuất một đối tượng phân số.

```
1. CPhanSo a;
```

```
2. a.Nhap();
```

```
3. a.Xuat();
```

## — Nhập, xuất một đối tượng phân số với thư viện `iostream`.

```
1. CPhanSo a;
```

```
2. cin>>a;
```

```
3. cout<<a;
```



Làm sao?

## 4. GIẢI QUYẾT VẤN ĐỀ

# 4. GIẢI QUYẾT VẤN ĐỀ

- Để giải quyết vấn đề trên ta phải định nghĩa
  - ✓ Toán tử vào (toán tử nhập): `operator>>` .
  - ✓ Toán tử ra (toán tử xuất): `operator <<` .
 cho lớp đối tượng `CPhanSo`.
- Ngoài ra, trong khi giải quyết vấn đề này ta còn sử dụng kỹ thuật hàm bạn (`friend function`) của phương pháp lập trình hướng đối tượng.
- Một “**hàm bạn**” của lớp đối tượng được phép truy xuất đến tất cả các thành phần của đối tượng thuộc về lớp đó bất chấp thành phần được khai báo trong phạm vi nào.

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:

21.};
    
```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
21.};

```

operator >>

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
                                operator >>
                                (istream &,                );
21.};

```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
                                operator >>
                                (istream &,CPhanSo &);

21.};
    
```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
                                istream& operator >>
                                (istream &,CPhanSo &);

21.};
    
```



# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
        friend istream& operator >>
            (istream &,CPhanSo &);
21.};
    
```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
        friend istream& operator >>
            (istream &,CPhanSo &);
            operator <<
21.};
    
```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        friend istream& operator >>
18.            (istream &,CPhanSo &);
19.        operator <<
20.            (ostream &,                );
21.};
    
```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
        friend istream& operator >>
            (istream &,CPhanSo &);
            operator <<
            (ostream &,CPhanSo &);
21.};
    
```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
        friend istream& operator >>
            (istream &,CPhanSo &);
        ostream& operator <<
            (ostream &,CPhanSo &);
21.};
    
```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
        friend istream& operator >>
            (istream &,CPhanSo &);
        friend ostream& operator <<
            (ostream &,CPhanSo &);
21.};
    
```

# 4. GIẢI QUYẾT VẤN ĐỀ

— Định nghĩa hàm toán tử vào (toán tử nhập):

```
11.istream& operator>>(istream &is,CPhanSo &x)
12.{
13.    cout << "Nhap tu:";
14.    is >> x.tu;
15.    cout << "Nhap mau:";
16.    is >> x.mau;
17.    return is;
18.}
```

Tại sao phải trả về một đối tượng thuộc lớp istream?

# 4. GIẢI QUYẾT VẤN ĐỀ

— Định nghĩa hàm toán tử ra (toán tử xuất):

```
11. ostream& operator<<(ostream &os, CPhanSo &x)
12. {
13.     os << "\n Tu: " << x.tu;
14.     os << "\n Mau: " << x.mau;
15.     return os;
16. }
```

Tại sao phải trả về một đối tượng thuộc lớp ostream?



# 5. HƯỚNG DẪN SỬ DỤNG 1

# 5. HƯỚNG DẪN SỬ DỤNG 1

- Hãy xem xét đoạn chương trình sau:
  1. CPhanSo a;
  2. `cin >> a;`
  3. cout << a;
- Trong câu lệnh thứ hai của đoạn chương trình trên ta nói: hàm **operator >>** được gọi thực hiện với 2 đối số là **cin** và đối tượng **a**.

# 5. HƯỚNG DẪN SỬ DỤNG 1

— Định nghĩa hàm toán tử vào (toán tử nhập):

```
11.istream& operator>>(istream &is,CPhanSo &x)
12.{
13.    cout << "Nhap tu:";
14.    is >> x.tu;
15.    cout << "Nhap mau:";
16.    is >> x.mau;
17.    return is;
18.}
```

Tại sao phải trả về một đối tượng thuộc lớp istream?

# 5. HƯỚNG DẪN SỬ DỤNG 1

- Hãy xem xét đoạn chương trình sau:
  1. `CPhanSo a;`
  2. `cin >> a;`
  3. `cout << a;`
- Trong câu lệnh thứ hai của đoạn chương trình trên ta nói: hàm **operator >>** được gọi thực hiện với 2 đối số là **cin** và đối tượng **a**.
- Trong câu lệnh thứ ba của đoạn chương trình trên ta nói: hàm **operator <<** được gọi thực hiện với 2 đối số là **cout** và đối tượng **a**.

# 5. HƯỚNG DẪN SỬ DỤNG 1

— Định nghĩa hàm toán tử ra (toán tử xuất):

```
11. ostream& operator<<(ostream &os, CPhanSo &x)
12. {
13.     os << "\n Tu: " << x.tu;
14.     os << "\n Mau: " << x.mau;
15.     return os;
16. }
```

Tại sao phải trả về một đối tượng thuộc lớp ostream?

# 6. HƯỚNG DẪN SỬ DỤNG 2

# 6. HƯỚNG DẪN SỬ DỤNG 2

- Hãy xem xét đoạn chương trình sau:

```
CPhanSo a,b,c;
```

```
cin >>a >>b >>c ;
```

```
cout<<a <<b <<c ;
```

# 6. HƯỚNG DẪN SỬ DỤNG 2

- Hãy xem xét đoạn chương trình sau:

```
CPhanSo a,b,c;
```

```
cin >>a >>b >>c ;
```

```
cout<<a <<b <<c ;
```



Giải  
thích



# 6. HƯỚNG DẪN SỬ DỤNG 2

— Định nghĩa hàm toán tử vào (toán tử nhập):

```
11.istream& operator>>(istream &is,CPhanSo &x)
12.{
13.    cout << "Nhap tu:";
14.    is >> x.tu;
15.    cout << "Nhap mau:";
16.    is >> x.mau;
17.    return is;
18.}
```

*(Note: In the original image, 'cin >> a' is circled and connected by a bubble to the question below.)*

Tại sao phải trả về một đối tượng thuộc lớp istream?

# 6. HƯỚNG DẪN SỬ DỤNG 2

- Hãy xem xét đoạn chương trình sau:

```
CPhanSo a,b,c;
```

```
cin >>a >>b >>c ;
```

```
cout<<a <<b <<c ;
```

**Giải  
thích**

# 6. HƯỚNG DẪN SỬ DỤNG 2

— Định nghĩa hàm toán tử vào (toán tử nhập):

```
11.istream& operator>>(istream &is,CPhanSo &x)
12.{
13.    cout << "Nhap tu:";
14.    is >> x.tu;
15.    cout << "Nhap mau:";
16.    is >> x.mau;
17.    return is;
18.}
```

`cin >> a >> b >> c ;`

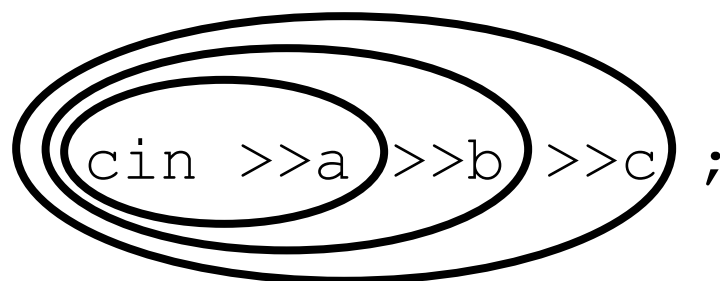
Tại sao phải trả về một đối tượng thuộc lớp istream?

# 6. HƯỚNG DẪN SỬ DỤNG 2

- Hãy xem xét đoạn chương trình sau:

```
CPhanSo a,b,c;
```

```
cin >>a >>b >>c ;
```



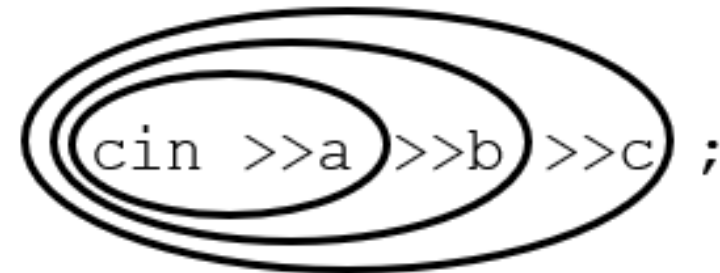
```
cout<<a <<b <<c ;
```

**Giải  
thích**

# 6. HƯỚNG DẪN SỬ DỤNG 2

— Định nghĩa hàm toán tử vào (toán tử nhập):

```
11.istream& operator>>(istream &is,CPhanSo &x)
12.{
13.    cout << "Nhap tu:";
14.    is >> x.tu;
15.    cout << "Nhap mau:";
16.    is >> x.mau;
17.    return is;
18.}
```



`cin >> a >> b >> c ;`

Tại sao phải trả về một đối tượng thuộc lớp istream?

# 6. HƯỚNG DẪN SỬ DỤNG 2

— Hãy xem xét đoạn chương trình sau:

```
CPhanSo a,b,c;
```

```
cin >>a >>b >>c ;
```

```
cout<<a <<b <<c ;
```

Giải  
thích

# 6. HƯỚNG DẪN SỬ DỤNG 2

— Định nghĩa hàm toán tử ra (toán tử xuất):

```
11. ostream& operator<<(ostream &os, CPhanSo &x)
12. {
13.     os << "\n Tu: " << x.tu;
14.     os << "\n Mau: " << x.mau;
15.     return os;
16. }
```

cout<<a<<b<<c ;

Tại sao phải trả về một đối tượng thuộc lớp ostream?

# 6. HƯỚNG DẪN SỬ DỤNG 2

— Hãy xem xét đoạn chương trình sau:

```
CPhanSo a,b,c;
```

```
cin >>a >>b >>c ;
```

```
cout<<a <<b <<c ;
```

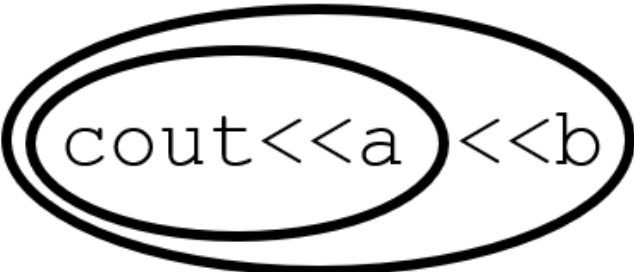
Giải  
thích



# 6. HƯỚNG DẪN SỬ DỤNG 2

— Định nghĩa hàm toán tử ra (toán tử xuất):

```
11. ostream& operator<<(ostream &os, CPhanSo &x)
12. {
13.     os << "\n Tu: " << x.tu;
14.     os << "\n Mau: " << x.mau;
15.     return os;
16. }
```

○ ○  <<c ;

Tại sao phải trả về một đối tượng thuộc lớp ostream?

# 6. HƯỚNG DẪN SỬ DỤNG 2

— Hãy xem xét đoạn chương trình sau:

```
CPhanSo a,b,c;
```

```
cin >>a >>b >>c ;
```

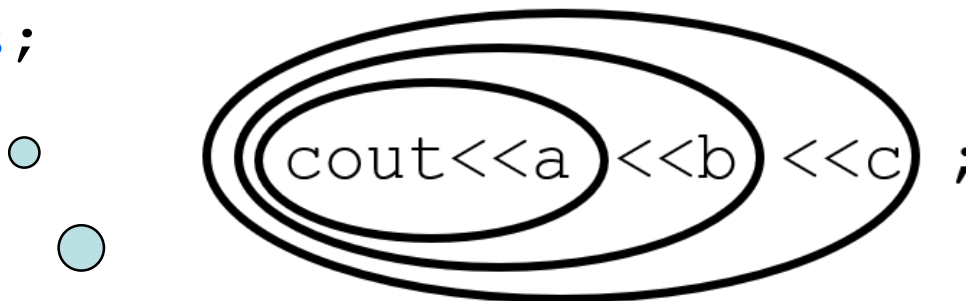
```
cout<<a <<b <<c ;
```

**Giải  
thích**

# 6. HƯỚNG DẪN SỬ DỤNG 2

— Định nghĩa hàm toán tử ra (toán tử xuất):

```
11. ostream& operator<<(ostream &os, CPhanSo &x)
12. {
13.     os << "\n Tu: " << x.tu;
14.     os << "\n Mau: " << x.mau;
15.     return os;
16. }
```



Tại sao phải trả về một đối tượng thuộc lớp ostream?

# 7. ỨNG DỤNG

## 7. ỨNG DỤNG

—Yêu cầu: Hãy định nghĩa toán tử vào và toán tử ra cho lớp đối tượng CNgay.

# 7. ỨNG DỤNG

```

11.class CNgay
12.{
13.    private:
14.        int ng;
15.        int th;
16.        int nm;
17.    public:
18.        friend istream& operator >>
19.            (istream &,CNgay &);
20.        friend ostream& operator <<
21.            (ostream &,CNgay &);
22.};
    
```

# 7. ỨNG DỤNG

— Định nghĩa hàm toán tử vào (toán tử nhập):

```

11.istream& operator>>(istream &is,CNgay &x)
12.{
13.    cout << "Nhap ngay:";
14.    is >> x.ng;
15.    cout << "Nhap thang:";
16.    is >> x.th;
17.    cout << "Nhap nam:";
18.    is >> x.nm;
19.    return is;
20.}
    
```

# 7. ỨNG DỤNG

— Định nghĩa hàm toán tử ra (toán tử xuất):

```

11. ostream& operator<<(ostream &os, CNgay &x)
12. {
13.     os <<"\n Ngay: " << x.ng;
14.     os <<"\n Thang: " << x.th;
15.     os <<"\n Nam: " << x.nm;
16.     return os;
17. }
    
```



# 8. BÀI TẬP VỀ NHÀ

# 8. BÀI TẬP VỀ NHÀ

— Hãy khai báo và định nghĩa hàm toán tử vào (*toán tử nhập*) và hàm toán tử ra (*toán tử xuất*) cho các lớp đối tượng sau:

1. Lớp điểm (**CDiem**).
2. Lớp điểm không gian (**CDiemKhongGian**).
3. Lớp phân số (**CPhanSo**).
4. Lớp hỗn số (**CHonSo**).
5. Lớp số phức (**CSoPhuc**).
6. Lớp ngày (**CNgay**).
7. Lớp thời gian (**CThoiGian**).
8. Lớp đơn thức (**CDonThuc**).

# 8. BÀI TẬP VỀ NHÀ

— Hãy khai báo và định nghĩa hàm toán tử vào và hàm toán tử ra cho các lớp đối tượng sau:

9. Lớp đường thẳng (**CDuongThang**) trong mặt phẳng Oxy.

10. Lớp đường tròn (**CDuongTron**) trong mặt phẳng Oxy.

11. Lớp lớp tam giác (**CTamGiac**) trong mặt phẳng Oxy.

12. Lớp hình cầu (**CHinhCau**) trong không gian Oxyz.