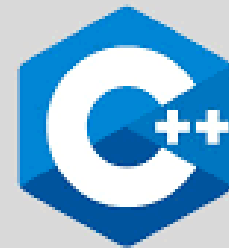


# Chương 4

## SƠ LƯỢC THƯ VIỆN IOSTREAM

- Nguyễn Hữu Lợi
- Đoàn Chánh Thống
- ThS. Nguyễn Thành Hiệp
- ThS. Trương Quốc Dũng
- ThS. Võ Duy Nguyên
- ThS. Nguyễn Văn Toàn
- TS. Nguyễn Duy Khánh
- TS. Nguyễn Tấn Trần Minh Khang

# 1. VÍ DỤ DẪN NHẬP 1



```
include  
<iostream>
```

# 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 << "Nhap mot so nguyen:";  
3. cin >> a;  
4. cout << "So nguyen vua nhap:" << a;
```



```
include  
<iostream>
```

# 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 << "Nhap mot so nguyen:";`

3. `cin >> a;`

4. `cout << "So nguyen vua nhap:" << a;`

Dòng 1 được đọc  
là:  $a$  là một biến có  
kiểu số nguyên `int`

# 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 << "Nhap mot so nguyen:";  
3. cin >> a;  
4. cout << "So nguyen vua nhap:" << a;
```

Dòng 2 được đọc là: toán tử xuất được gọi thực hiện với hai đối số là `cout` và chuỗi thông báo “Nhap mot so nguyen”

# 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 << "Nhap mot so nguyen:";`

3. `cin >> a;`

4. `cout << "So nguyen vua nhap:" << a;`

Dòng 3 được đọc là: toán tử nhập được gọi thực hiện với hai đối số là `cin` và biến  $a$ .



# 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`. Dòng 4 được đọc là: toán tử xuất được gọi thực hiện với hai đối số là `cout` và chuỗi thông báo ... và toán tử xuất được tiếp tục nạp chồng với đối số là biến  $a$ .

1. `int a;`
2. `cout << "Nhap mot so nguyen:";`
3. `cin >> a;`
4. `cout << "So nguyen vua nhap:" << a;`

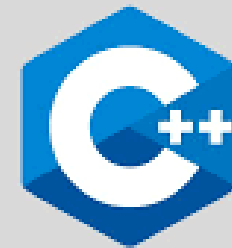


# 1. Ví dụ dẫn nhập 1

— Nhắc lại:

- + Ký hiệu `>>` được gọi là toán tử vào, toán tử nhập.
- + Ký hiệu `<<` được gọi là toán tử ra, toán tử xuất.
- + `cin` là đối tượng thuộc lớp đối tượng `istream`.
- + `cout` là đối tượng thuộc lớp đối tượng `ostream`.

## 2. VÍ DỤ DẪN NHẬP 2



```
include  
<iostream>
```

## 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;
```



```
include  
<iostream>
```

## 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. }
```



```
include  
<iostream>
```

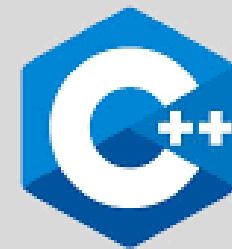
## 2. Ví dụ dẫn nhập 2

```
18. void Xuat(PHANSO x)
19. {
20. |   cout << x.Tu << "/" << x.Mau;
21. }
```



```
include  
<iostream>
```

### 3. ĐẶT VẤN ĐỀ



```
include  
<iostream>
```

## 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;     •     •     •

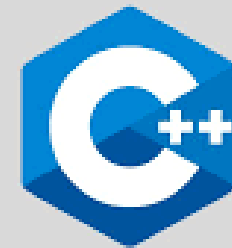
3. cout << a;



Làm sao?



## 4. GIẢI QUYẾT VẤN ĐỀ



```
include  
<iostream>
```

## 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” (`friend function`) 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:
17.        friend ostream& operator>>(ostream &,CPhanSo &);
18.        friend ostream& operator<<(ostream &,CPhanSo &);
19.};
```

## 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>>(istream &,CPhanSo &);
18.        friend ostream& operator<<(ostream &,CPhanSo &);
19.};
```

## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 4. Giải quyết vấn đề

— Khai báo lớp.

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



## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 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>>(istream &,CPhanSo &);
18.        friend ostream& operator<<(ostream &,CPhanSo &);
19.};
```

## 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>>(istream &,CPhanSo &);
18.        friend ostream& operator<<(ostream &,CPhanSo &);
19.};
```

## 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>>(istream &,CPhanSo &);
18.        friend ostream& operator<<(ostream &,CPhanSo &);
19.};
```



## 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>>(istream &,CPhanSo &);
18.        friend ostream& operator<<(ostream &,CPhanSo &);
19.};
```

## 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>>(istream &,CPhanSo &);
18.        friend ostream& operator<<(ostream &,CPhanSo &);
19.};
```

## 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>>(istream &,CPhanSo &);
18.        friend ostream& operator<<(ostream &,CPhanSo &);
19.};
```

## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 4. Giải quyết vấn đề

— Khai báo lớp.

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

## 4. Giải quyết vấn đề

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

```
20. istream& operator >> (istream &is, CPhanSo &x)
21. {
22.     cout << "Nhap tu:";
23.     is >> x.Tu;
24.     cout << "Nhap mau:";
25.     is >> x.Mau;
26.     return is;
27. }
```

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



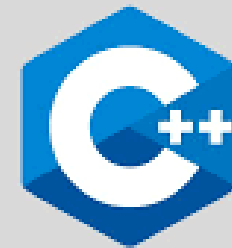
## 4. Giải quyết vấn đề

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

```
28. ostream& operator << (ostream &os, CPhanSo &x)  
29. {  
30.     os << "\n Tu: " << x.Tu;  
31.     os << "\n Mau: " << x.Mau;  
32.     return os;  
33. }
```

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

## 5. HƯỚNG DẪN SỬ DỤNG 1



```
include  
<iostream>
```

## 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):

```
20. istream& operator >> (istream &is, CPhanSo &x)
21. {
22.     cout << "Nhap tu:";
23.     is >> x.Tu;
24.     cout << "Nhap mau:";
25.     is >> x.Mau;
26.     return is;
27. }
```

## 5. Hướng dẫn sử dụng 1

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

```
20. istream& operator >> (istream &is, CPhanSo &x)
21. {
22.     cout << "Nhap tu:";
23.     is >> x.Tu;
24.     cout << "Nhap mau:";
25.     is >> x.Mau;
26.     return is;
27. }
```

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 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):

```
28. ostream& operator << (ostream &os, CPhanSo &x)
29. {
30.     os << "\n Tu: " << x.Tu;
31.     os << "\n Mau: " << x.Mau;
32.     return os;
33. }
```



# 5. HƯỚNG DẪN SỬ DỤNG 1

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

```
28. ostream& operator << (ostream &os, CPhanSo &x)  
29. {  
30.     os << "\n Tu: " << x.Tu;  
31.     os << "\n Mau: " << x.Mau;  
32.     return os;  
33. }
```

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



## 6. HƯỚNG DẪN SỬ DỤNG 2



```
include  
<iostream>
```

## 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 ;
```

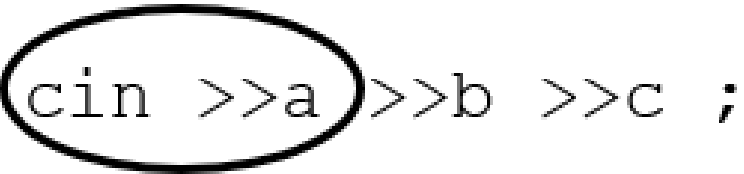


Giải thích

## 6. Hướng dẫn sử dụng 2

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

```
20. istream& operator >> (istream &is, CPhanSo &x)
21. {
22.     cout << "Nhap tu:";
23.     is >> x.Tu;
24.     cout << "Nhap mau:";
25.     is >> x.Mau;
26.     return is;
27. }
```



## 6. Hướng dẫn sử dụng 2

— Hàm toán tử vào được gọi thực hiện với hai đối số:

- + Đối số thứ nhất là đối tượng `cin`.
- + Đối số thứ hai là đối tượng `a`.

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

— Vào bên trong hàm toán tử vào:

- + Đối số thứ nhất tương ứng với đối tượng `is`.
- + Đối số thứ hai tương ứng đối tượng `x`.

```
20. istream& operator >> (istream &is, CPhanSo &x)
21. {
22.     cout << "Nhap tu:";
23.     is >> x.Tu;
24.     cout << "Nhap mau:";
25.     is >> x.Mau;
26.     return is;
27. }
```

## 6. Hướng dẫn sử dụng 2

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

```

20. istream& operator >> (istream &is, CPhanSo &x)
21. {
22.     cout << "Nhap tu:";
23.     is >> x.Tu;
24.     cout << "Nhap mau:";
25.     is >> x.Mau;
26.     return is;
27. }
  
```

*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):

```
20. istream& operator >> (istream &is, CPhanSo &x)
```

```
21. {
```

```
22.     cout << "Nhap tu:";
```

```
23.     is >> x.Tu;
```

```
24.     cout << "Nhap mau:";
```

```
25.     is >> x.Mau;
```

```
26.     return is;
```

```
27. }
```



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



## 6. Hướng dẫn sử dụng 2

— Hàm toán tử vào được gọi thực hiện với hai đối số:

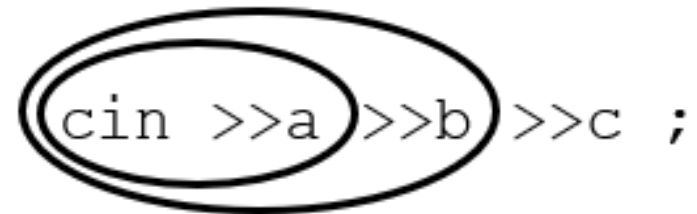
+ Đối số thứ nhất là đối tượng trả về của lần gọi thực hiện toán tử vào thứ nhất (đối tượng **cin**).

+ Đối số thứ hai là đối tượng **b**.

— Vào bên trong hàm toán tử vào:

+ Đối số thứ nhất tương ứng với đối tượng **is**.

+ Đối số thứ hai tương ứng đối tượng **x**.



```

20. istream& operator >> (istream &is, CPhanSo &x)
21. {
22.     cout << "Nhap tu:";
23.     is >> x.Tu;
24.     cout << "Nhap mau:";
25.     is >> x.Mau;
26.     return is;
27. }
  
```

## 6. Hướng dẫn sử dụng 2

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

```
20. istream& operator >> (istream &is, CPhanSo &x)
```

```
21. {
```

```
22.     cout << "Nhap tu:";
```

```
23.     is >> x.Tu;
```

```
24.     cout << "Nhap mau:";
```

```
25.     is >> x.Mau;
```

```
26.     return is;
```

```
27. }
```

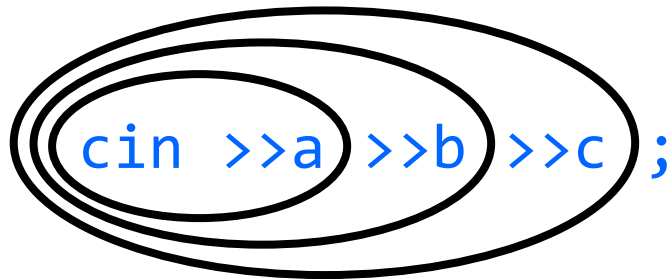
`cin >> a >> b >> c ;`

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):

```
20. istream& operator >> (istream &is, CPhanSo &x)
```

```
21. {
```

```
22.     cout << "Nhap tu:";
```

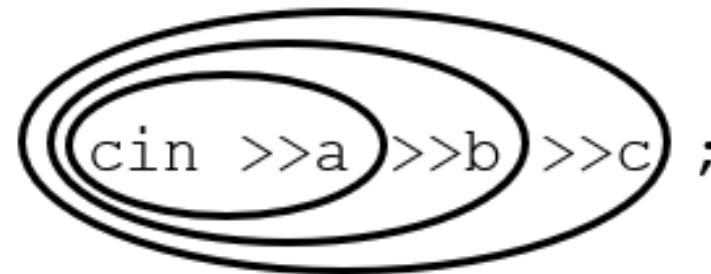
```
23.     is >> x.Tu;
```

```
24.     cout << "Nhap mau:";
```

```
25.     is >> x.Mau;
```

```
26.     return is;
```

```
27. }
```



## 6. Hướng dẫn sử dụng 2

— Hàm toán tử vào được gọi thực hiện với hai đối số:

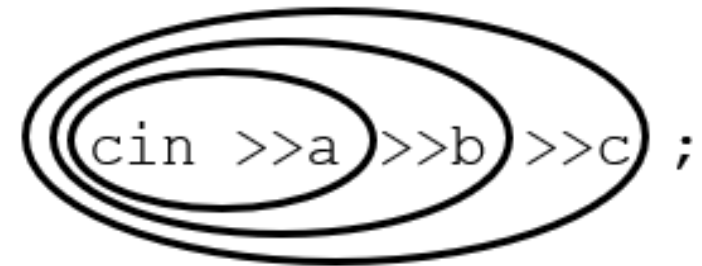
+ Đối số thứ nhất là đối tượng trả về của lần gọi thực hiện toán tử vào thứ hai (đối tượng **cin**).

+ Đối số thứ hai là đối tượng **c**.

— Vào bên trong hàm toán tử vào:

+ Đối số thứ nhất tương ứng với đối tượng **is**.

+ Đối số thứ hai tương ứng đối tượng **x**.



```

20.istream& operator >> (istream &is, CPhanSo &x)
21.{
22.    cout << "Nhap tu:";
23.    is >> x.Tu;
24.    cout << "Nhap mau:";
25.    is >> x.Mau;
26.    return is;
27.}
    
```

## 6. Hướng dẫn sử dụng 2

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

```
20. istream& operator >> (istream &is, CPhanSo &x)
```

```
21. {
```

```
22.     cout << "Nhap tu:";
```

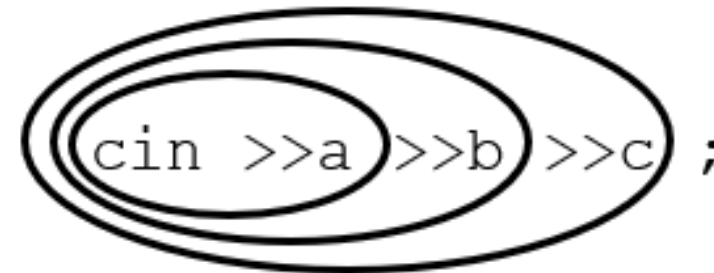
```
23.     is >> x.Tu;
```

```
24.     cout << "Nhap mau:";
```

```
25.     is >> x.Mau;
```

```
26.     return is;  ° ° °
```

```
27. }
```



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):

```
28. ostream& operator << (ostream &os, CPhanSo &x)
```

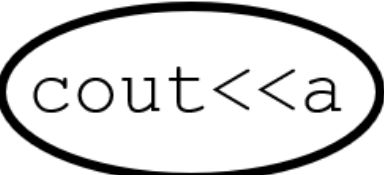
```
29. {
```

```
30. |   os << "\n Tu: " << x.Tu;
```

```
31. |   os << "\n Mau: " << x.Mau;
```

```
32. |   return os;
```

```
33. }
```

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



## 6. Hướng dẫn sử dụng 2

— Hàm toán tử ra được gọi thực hiện với hai đối số:

- + Đối số thứ nhất là đối tượng `cout`.
- + Đối số thứ hai là đối tượng `a`.

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

— Vào bên trong hàm toán tử ra:

- + Đối số thứ nhất tương ứng với đối tượng `os`.
- + Đối số thứ hai tương ứng đối tượng `x`.

```
28. ostream& operator << (ostream &os, CPhanSo &x)
29. {
30.     os << "\n Tu: " << x.Tu;
31.     os << "\n Mau: " << x.Mau;
32.     return os;
33. }
```

## 6. Hướng dẫn sử dụng 2

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

```
28. ostream& operator << (ostream &os, CPhanSo &x)
```

```
29. {
```

```
30. |   os << "\n Tu: " << x.Tu;
```

```
31. |   os << "\n Mau: " << x.Mau;
```

```
32. |   return os;
```

```
33. }
```

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):

```
28. ostream& operator << (ostream &os, CPhanSo &x)
```

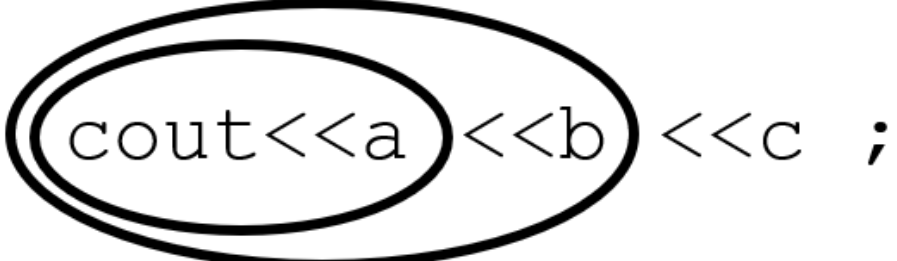
```
29. {
```

```
30. |   os << "\n Tu: " << x.Tu;
```

```
31. |   os << "\n Mau: " << x.Mau;
```

```
32. |   return os;
```

```
33. }
```



cout << a << b << c ;

## 6. Hướng dẫn sử dụng 2

— Hàm toán tử ra được gọi thực hiện với hai đối số:

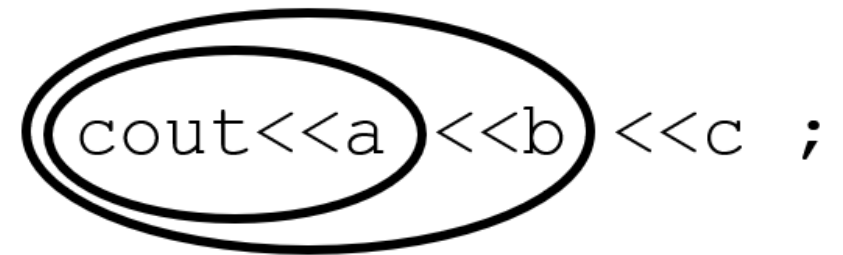
+ Đối số thứ nhất là đối tượng trả về của lần gọi thực hiện toán tử ra thứ nhất (đối tượng **cout**).

+ Đối số thứ hai là đối tượng **a**.

— Vào bên trong hàm toán tử ra:

+ Đối số thứ nhất tương ứng với đối tượng **os**.

+ Đối số thứ hai tương ứng đối tượng **x**.



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

```
28. ostream& operator << (ostream &os, CPhanSo &x)
29. {
30.     os << "\n Tu: " << x.Tu;
31.     os << "\n Mau: " << x.Mau;
32.     return os;
33. }
```

## 6. Hướng dẫn sử dụng 2

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

```
28. ostream& operator << (ostream &os, CPhanSo &x)
```

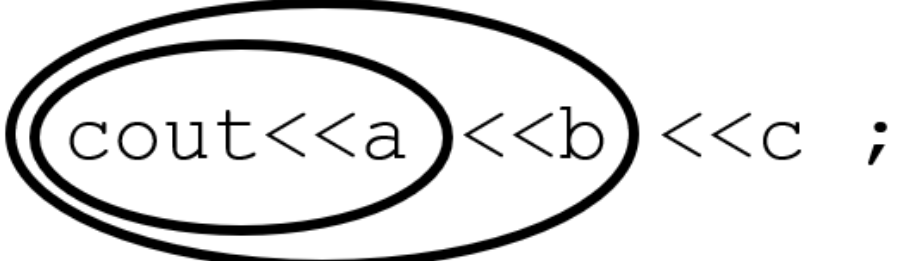
```
29. {
```

```
30.     os << "\n Tu: " << x.Tu;
```

```
31.     os << "\n Mau: " << x.Mau;
```

```
32.     return os;
```

```
33. }
```



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):

```
28. ostream& operator << (ostream &os, CPhanSo &x)
```

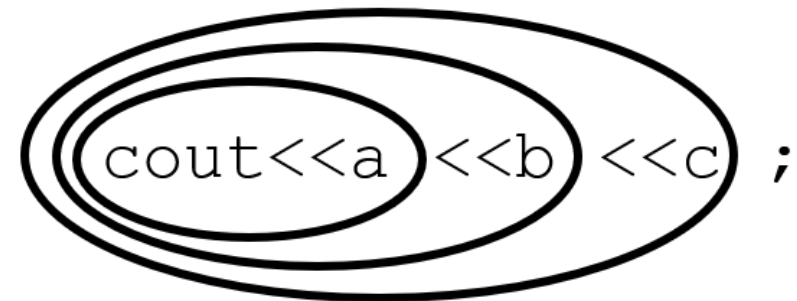
```
29. {
```

```
30. |   os << "\n Tu: " << x.Tu;
```

```
31. |   os << "\n Mau: " << x.Mau;
```

```
32. |   return os;
```

```
33. }
```





## 6. Hướng dẫn sử dụng 2

— Hàm toán tử ra được gọi thực hiện với hai đối số:

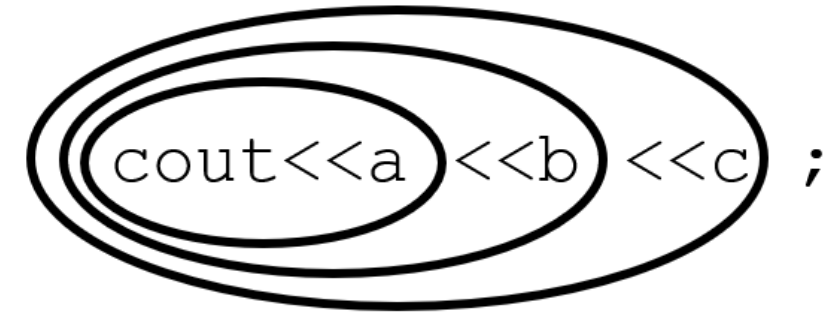
+ Đối số thứ nhất là đối tượng trả về của lần gọi thực hiện toán tử ra thứ hai (đối tượng **cout**).

+ Đối số thứ hai là đối tượng **a**.

— Vào bên trong hàm toán tử ra:

+ Đối số thứ nhất tương ứng với đối tượng **os**.

+ Đối số thứ hai tương ứng đối tượng **x**.



```

28. ostream& operator << (ostream &os, CPhanSo &x)
29. {
30.     os << "\n Tu: " << x.Tu;
31.     os << "\n Mau: " << x.Mau;
32.     return os;
33. }
  
```

## 6. Hướng dẫn sử dụng 2

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

```
28. ostream& operator << (ostream &os, CPhanSo &x)
```

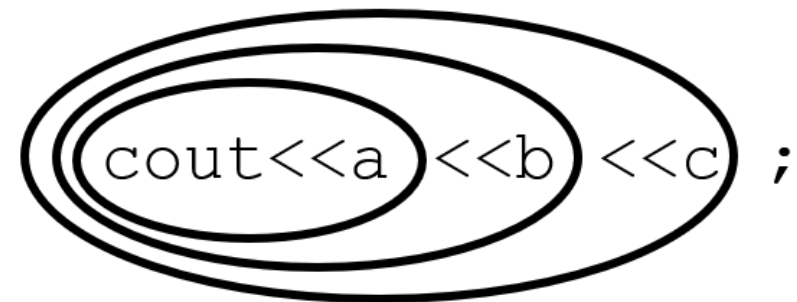
```
29. {
```

```
30.     os << "\n Tu: " << x.Tu;
```

```
31.     os << "\n Mau: " << x.Mau;
```

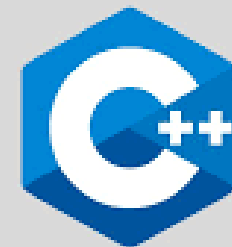
```
32.     return os;
```

```
33. }
```



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

## 7. ỨNG DỤNG



```
include  
<iostream>
```

## 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 Ngay;
15.        int Thang;
16.        int Nam;
17.    public:
18.        friend ostream& operator >>(ostream &,CNgay &);
19.        friend ostream& operator <<(ostream &,CNgay &);
20.};
```

## 7. Ứng dụng

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

```
21. istream& operator >> (istream &is, CNgay &x)
22. {
23.     cout << "Nhap ngay:";
24.     is >> x.Ngay;
25.     cout << "Nhap thang:";
26.     is >> x.Thang;
27.     cout << "Nhap nam:";
28.     is >> x.Nam;
29.     return is;
30. }
```

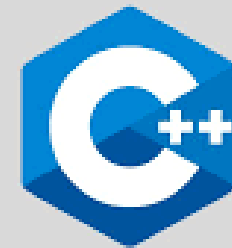
## 7. Ứng dụng

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

```
31. ostream& operator << (ostream &os, CNgay &x)
32. {
33.     os << "\n Ngay: " << x.Ngay;
34.     os << "\n Thang: " << x.Thang;
35.     os << "\n Nam: " << x.Nam;
36.     return os;
37. }
```



## 8. BÀI TẬP VỀ NHÀ



```
include  
<iostream>
```

## 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).

## 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:

7. Lớp thời gian (CThoiGian).

8. Lớp đơn thức (CDonThuc).

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

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

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

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

## 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:

13.Lớp mảng một chiều tĩnh (CMangTinh).

14.Lớp mảng một chiều động (CMangDong).

15.Lớp ma trận tĩnh (CMaTranTinh).

16.Lớp ma trận động (CMaTranDong).

17.Lớp đa thức tĩnh (CDaThucTinh).

18.Lớp đa thức động (CDaThucDong).

**Cảm ơn quý vị đã lắng nghe**

**Nhóm tác giả**

**TS. Nguyễn Tấn Trần Minh Khang**

# Chương 4

## PHƯƠNG THỨC XUẤT TRONG PYTHON

- Nguyễn Hữu Lợi
- Đoàn Chánh Thống
- ThS. Nguyễn Thành Hiệp
- ThS. Trương Quốc Dũng
- ThS. Võ Duy Nguyên
- ThS. Nguyễn Văn Toàn
- TS. Nguyễn Duy Khánh
- TS. Nguyễn Tấn Trần Minh Khang

# 1. ĐẶT VẤN ĐỀ – PYTHON



# 1. Đặt vấn đề

— Nhập xuất một đối tượng phân số.

1. `a = CPhanSo()`

2. `a.Nhap()`

3. `a.Xuat()`

— Xuất một đối tượng phân số với câu lệnh `print`.

1. `a = CPhanSo()`

2. `a.Nhap()`

3. `print(a)`



Làm sao?

## 2. GIẢI QUYẾT VẤN ĐỀ – PYTHON

## 2. Giải quyết vấn đề

- Để giải quyết vấn đề trên ta phải định nghĩa phương thức `__str__` cho lớp đối tượng CPhanSo.

## 2. Giải quyết vấn đề

```
1. class CPhanSo:
2.     def __init__(self):
3.         self.Tu = 0
4.         self.Mau = 1
5.     def Nhap(self):
6.         self.Tu = int(input("Nhap tu: "))
7.         self.Mau = int(input("Nhap mau: "))
8.     ...
```

## 2. Giải quyết vấn đề

```
1. class CPhanSo:
2.     ...
9.     def Xuat(self):
10.         print(self.Tu, "/", self.Mau)

11.     def __str__(self):
12.         return f"{self.Tu} / {self.Mau}"
```

## 2. Giải quyết vấn đề

```
13. def main():  
14.     ps = CPhanSo()  
15.     ps.Nhap()  
16.     ps.Xuat()  
17.     print(ps)  
18.  
19. if __name__ == "__main__":  
20.     main()
```

## 3. BÀI TẬP VỀ NHÀ – PYTHON



## 3. Bài tập về nhà

— Hãy định nghĩa phương thức `__str__` 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. ...

### 3. Bài tập về nhà

— Hãy định nghĩa phương thức `__str__` cho các lớp đối tượng sau:

...

7. Lớp thời gian (CThoiGian).

8. Lớp đơn thức (CDonThuc).

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

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

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

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

**Cảm ơn quý vị đã lắng nghe**

**Nhóm tác giả**

**TS. Nguyễn Tấn Trần Minh Khang**