

# Chương 5

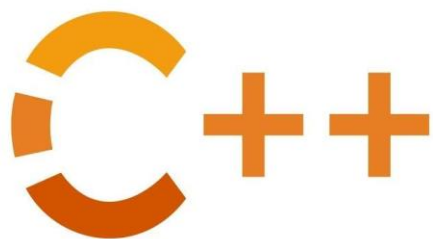
## PHƯƠNG THỨC THIẾT LẬP, PHƯƠNG THỨC PHÁ HỦY

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

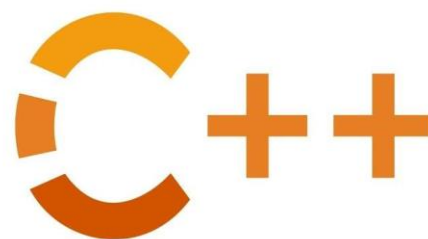
# 1. MỤC TIÊU

# 1. MỤC TIÊU

- Hiểu được phương thức thiết lập (**constructors**) là gì?
- Hiểu được phương thức phá hủy (**destructor**) là gì?



Constructors



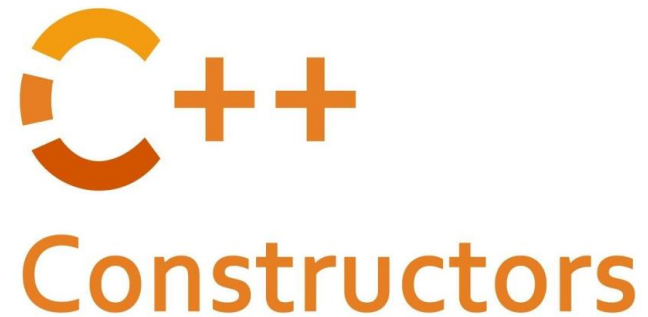
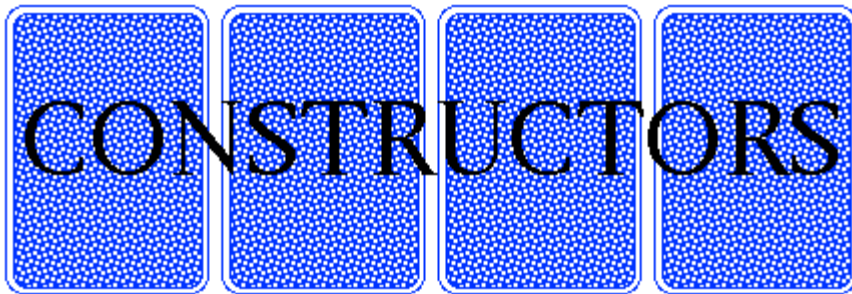
Destructors

Constructors

## 2. PHƯƠNG THỨC THIẾT LẬP

## 2. PHƯƠNG THỨC THIẾT LẬP

- **Mục tiêu:** các *phương thức thiết lập* (*constructors*) của một *lớp đối tượng* (*class*) có nhiệm vụ thiết lập thông tin ban đầu cho các đối tượng thuộc về lớp ngay khi đối tượng được khai báo.

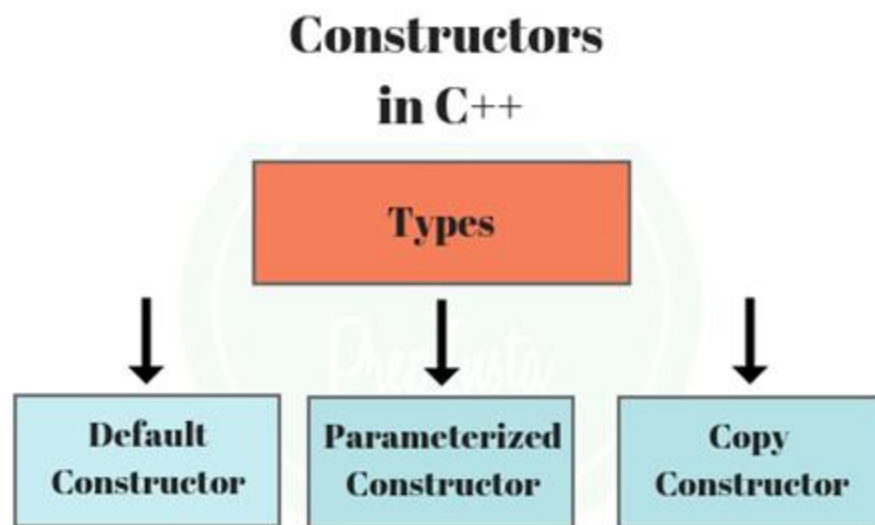


## 2. PHƯƠNG THỨC THIẾT LẬP

- Các đặc điểm của phương thức thiết lập (constructors):
  - + Tên phương thức thiết lập trùng với tên lớp.
  - + Không có giá trị trả về.
  - + Được tự động gọi thực hiện ngay khi đối tượng được khai báo.
  - + Có thể có nhiều phương thức thiết lập trong 1 lớp.
  - + Trong một quá trình sống của đối tượng thì chỉ có 1 lần duy nhất một phương thức thiết lập được gọi thực hiện mà thôi đó là khi đối tượng ra đời.
  - + Các phương thức thiết lập của lớp thuộc nhóm các phương thức khởi tạo.

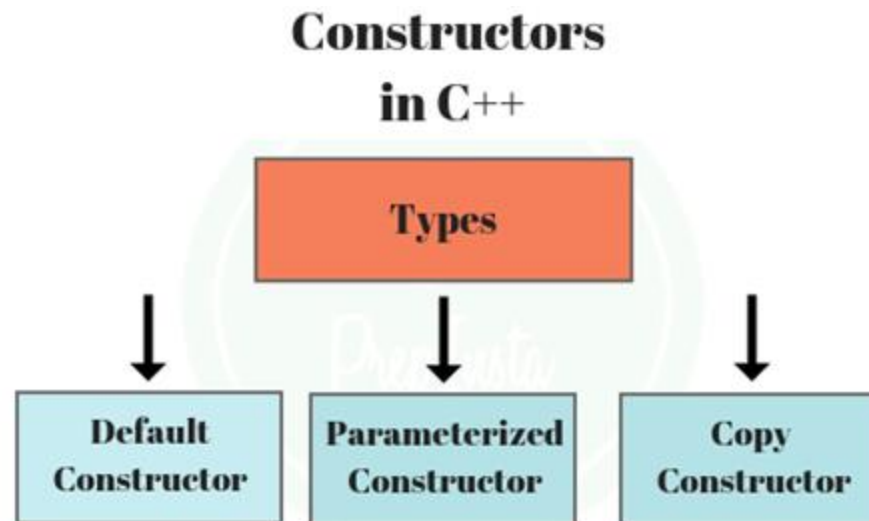
## 2. PHƯƠNG THỨC THIẾT LẬP

- **Phân loại phương thức thiết lập:** Ta có thể chia các phương thức thiết lập của một lớp thành 3 nhóm như sau:
  - + Phương thức thiết lập mặc định (*default constructor*).
  - + Phương thức thiết lập sao chép (*copy constructor*).
  - + Phương thức thiết lập nhận tham số đầu vào (*user define constructor – parameterized constructors*).



# 2. PHƯƠNG THỨC THIẾT LẬP

- Phương thức thiết lập mặc định (*default constructor*) là: phương thức thiết lập các thông tin ban đầu cho đối tượng thuộc về lớp bằng những giá trị mặc định (do người lập trình quyết định).

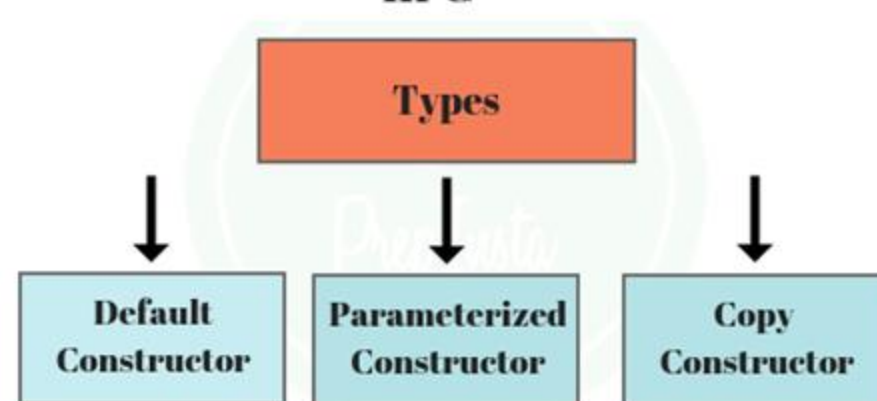




## 2. PHƯƠNG THỨC THIẾT LẬP

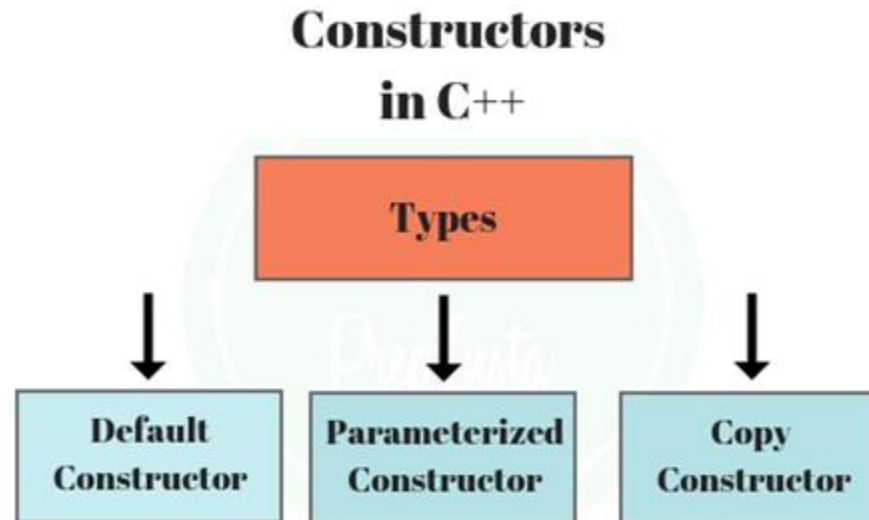
- Phương thức thiết lập sao chép (*copy constructor*) là: phương thức thiết lập nhận tham số đầu vào là *một đối tượng cùng thuộc về lớp*.
  - + Các thông tin ban đầu của đối tượng sẽ hoàn toàn giống thông tin của đối tượng tham số đầu vào.
  - + Ngoài ra, người ta còn nói phương thức thiết lập sao chép được sử dụng để tạo ra đối tượng mới giống hoàn toàn đối tượng đã có sẵn.

### Constructors in C++



## 2. PHƯƠNG THỨC THIẾT LẬP

- Phương thức thiết lập nhận tham số đầu vào là (*user define constructor – parameterized constructors*): những phương thức thiết lập ko phải là phương thức thiết lập mặc định và phương thức thiết lập sao chép.



## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.
- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        CPhanSo();
18.        CPhanSo(int, int);
19.        CPhanSo(const CPhanSo&);
20.};


```

## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.
- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        CPhanSo();
18.        CPhanSo(int, int);
19.        CPhanSo(const CPhanSo&);
20.};
    
```




Phương thức thiết  
lập mặc định

## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.
- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        CPhanSo();
18.        CPhanSo(int, int);
19.        CPhanSo(const CPhanSo&);
20.};
    
```




Phương thức thiết  
lập sao chép

## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.
- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        CPhanSo();
18.        CPhanSo(int, int);
19.        CPhanSo(const CPhanSo&);
20.};
    
```




Phương thức thiết lập khi biết đầy đủ thông tin

## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.
- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        CPhanSo();
18.        CPhanSo(int, int);
19.        CPhanSo(const CPhanSo&);
20.};
    
```



Tên phương thức  
thiết lập trùng với  
tên lớp.

## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.
- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        CPhanSo();
18.        CPhanSo(int, int);
19.        CPhanSo(const CPhanSo&);
20.};

```



Không có giá trị  
trả về.



## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.

- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        CPhanSo();
18.        CPhanSo(int, int);
19.        CPhanSo(const CPhanSo&);
20.};
    
```

Có thể có  
nhiều phương  
thức thiết lập  
trong 1 lớp.

## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.

- Khai báo lớp.

```

11. class CPhanSo
12. {
13.     private:
14.         int tu;
15.         int mau;
16.     public:
17.         CPhanSo();
18.         CPhanSo(int, int);
19.         CPhanSo(const CPhanSo&);
20. };
    
```

Các phương thức thiết lập của lớp thuộc nhóm các phương thức khởi tạo.

## 2. PHƯƠNG THỨC THIẾT LẬP

- Các đặc điểm của phương thức thiết lập (constructors):
  - + Tên phương thức thiết lập trùng với tên lớp.
  - + Không có giá trị trả về.
  - + Được tự động gọi thực hiện ngay khi đối tượng được khai báo.
  - + Có thể có nhiều phương thức thiết lập trong 1 lớp.
  - + Trong một quá trình sống của đối tượng thì chỉ có 1 lần duy nhất một phương thức thiết lập được gọi thực hiện mà thôi đó là khi đối tượng ra đời.
  - + Các phương thức thiết lập của lớp thuộc nhóm các phương thức khởi tạo.

## 2. PHƯƠNG THỨC THIẾT LẬP

- Ví dụ minh họa: Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng CPhanSo.
- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        CPhanSo();
18.        CPhanSo(int, int);
19.        CPhanSo(const CPhanSo&);
20.};

```

## 2. PHƯƠNG THỨC THIẾT LẬP

- Định nghĩa phương thức thiết lập mặc định.
- Phương thức thiết lập mặc định (*default constructor*) là: phương thức thiết lập các thông tin ban đầu cho đối tượng thuộc về lớp bằng những giá trị mặc định (do người lập trình quyết định).

```
11.CPhanSo::CPhanSo()
```

```
12.{
```

```
13.    tu = 0;
```

```
14.    mau = 1;
```

```
15.}
```

## 2. PHƯƠNG THỨC THIẾT LẬP

- Định nghĩa phương thức thiết lập sao chép.
- Phương thức thiết lập sao chép (*copy constructor*) là: phương thức thiết lập nhận tham số đầu vào là *một đối tượng cùng thuộc về lớp*.

```
11.CPhanSo::CPhanSo(const CPhanSo&x)
```

```
12.{
```

```
13.    |    tu = x.tu;
```

```
14.    |    mau = x.mau;
```

```
15.}
```

## 2. PHƯƠNG THỨC THIẾT LẬP

- Định nghĩa phương thức thiết lập khi biết đầy đủ thông tin.

```

11.CPhanSo::CPhanSo(int t,int m)
12.{
13.    |    tu = t;
14.    |    mau = m;
15.}
    
```

## 2. PHƯƠNG THỨC THIẾT LẬP

- **Hướng dẫn sử dụng 01:** Hãy xem xét đoạn chương trình sau và cho biết có bao nhiêu phương thức gọi thực hiện:
  - 11.....
  - 12.CPhanSo a;
  - 13.a.Nhap();
  - 14.a.Xuat();
- Trả lời: Có      phương thức được gọi thực hiện.
  - + Đối tượng a gọi thực hiện phương thức....
  - + Đối tượng a gọi thực hiện phương thức nhập.
  - + Đối tượng a gọi thực hiện phương thức xuất.



## 2. PHƯƠNG THỨC THIẾT LẬP

- Các đặc điểm của phương thức thiết lập (constructors):
  - + Tên phương thức thiết lập trùng với tên lớp.
  - + Không có giá trị trả về.
  - + Được tự động gọi thực hiện ngay khi đối tượng được khai báo.
  - + Có thể có nhiều phương thức thiết lập trong 1 lớp.
  - + Trong một quá trình sống của đối tượng thì chỉ có 1 lần duy nhất một phương thức thiết lập được gọi thực hiện mà thôi đó là khi đối tượng ra đời.
  - + Các phương thức thiết lập của lớp thuộc nhóm các phương thức khởi tạo.

## 2. PHƯƠNG THỨC THIẾT LẬP

– **Hướng dẫn sử dụng 02:**

Hãy xem xét đoạn chương trình sau và cho biết có bao nhiêu phương thức gọi thực hiện:

```
1. CPhanSo a;  
2. CPhanSo b(1, 2);  
3. a.Nhap();  
4. b.Xuat();  
5. CPhanSo c(a);  
6. c.Xuat();
```

– Trả lời: Có      phương thức được gọi thực hiện.

- + Phương thức....
- + Phương thức....
- + Phương thức....
- + Phương thức....
- + Phương thức....
- + Phương thức....

## 2. PHƯƠNG THỨC THIẾT LẬP

- Ý nghĩa việc sử dụng phương thức thiết lập:
  - + Khởi tạo giá trị ban đầu cho các đối tượng thuộc về lớp ngay khi các đối tượng được khai báo.
  - + Ép kiểu từ đối tượng này sang đối tượng khác.

# 2. PHƯƠNG THỨC THIẾT LẬP

— **Áp dụng:** Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản cho lớp đối tượng ngày.

— Khai báo lớp.

```

11.class CNgay
12.{
13.    private:
14.        int ng;
15.        int th;
16.        int nm;
17.    public:
18.        CNgay();
19.        CNgay(const CNgay &);
20.        CNgay(int,int,int);
21.};
    
```

# 2. PHƯƠNG THỨC THIẾT LẬP

— Định nghĩa phương thức thiết lập mặc định.

```
11.CNgay::CNgay()
```

```
12.{
```

```
13.    |    ng = 1;
```

```
14.    |    th = 1;
```

```
15.    |    nm = 1;
```

```
16.}
```

# 2. PHƯƠNG THỨC THIẾT LẬP

— Định nghĩa phương thức thiết lập sao chép.

```
11. CNgay::CNgay(const CNgay&x)
```

```
12. {
```

```
13.     |   ng = x.ng;
```

```
14.     |   th = x.th;
```

```
15.     |   nm = x.nm;
```

```
16. }
```

# 2. PHƯƠNG THỨC THIẾT LẬP

- Định nghĩa phương thức thiết lập khi biết đầy đủ thông tin.

```

11. CNgay::CNgay(int ngng, int thth, int nmnm)
12. {
13.     ng = ngng;
14.     th = thth;
15.     nm = nmnm;
16. }
    
```

Destructor

## 3. PHƯƠNG THỨC PHÁ HỦY



# 3. PHƯƠNG THỨC PHÁ HỦY

- **Mục tiêu:** Phương thức phá hủy (*destructor*) của một lớp có nhiệm vụ dọn dẹp “*xác chết*” của đối tượng khi đối tượng “*đi bán muối*”. Nói một cách khác, phương thức phá hủy có nhiệm vụ thu hồi lại tất cả các tài nguyên đã cấp phát cho đối tượng khi đối tượng hết *phạm vi hoạt động* (scope)



## Destructors

# 3. PHƯƠNG THỨC PHÁ HỦY

- Đặc điểm của phương thức phá hủy:
  - + Tên phương thức trùng với tên lớp nhưng có dấu ngã ở đằng trước.
  - + Không có giá trị trả về.
  - + Không có tham số đầu vào.
  - + Được tự động gọi thực hiện khi đối tượng hết phạm vi sử dụng.
  - + ...

# 3. PHƯƠNG THỨC PHÁ HỦY

— Đặc điểm của phương thức phá hủy:

+ ...

+ Phương thức phá hủy thuộc nhóm các phương thức xử lý.

+ Có và chỉ có duy nhất một phương thức phá hủy trong 1 lớp mà thôi.

+ Trong một quá trình sống của đối tượng có và chỉ có một lần phương thức phá hủy được gọi thực hiện mà thôi.

# 3. PHƯƠNG THỨC PHÁ HỦY

## — Đặc điểm của phương thức phá hủy:

- + Tên phương thức trùng với tên lớp nhưng có dấu ngã ở đằng trước.
- + Không có giá trị trả về.
- + Không có tham số đầu vào.
- + Được tự động gọi thực hiện khi đối tượng hết phạm vi sử dụng.
- + Phương thức phá hủy thuộc nhóm các phương thức xử lý.
- + Có và chỉ có duy nhất một phương thức phá hủy trong 1 lớp mà thôi.
- + Trong một quá trình sống của đối tượng có và chỉ có một lần phương thức phá hủy được gọi thực hiện mà thôi.

# 3. PHƯƠNG THỨC PHÁ HỦY

- Ví dụ minh họa: Hãy khai báo và định nghĩa phương thức phá hủy cho lớp đối tượng CPhanSo.

- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        // Nhóm phương thức xử lý
18.        ~CPhanSo ();
19.} ;
    
```

# 3. PHƯƠNG THỨC PHÁ HỦY

- Ví dụ minh họa: Hãy khai báo và định nghĩa phương thức phá hủy cho lớp đối tượng CPhanSo.

- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        // Nhóm phương thức và biến
18.        ~CPhanSo ();
19.} ;
    
```

Tên phương thức  
trùng với tên lớp  
nhưng có dấu ngã ở  
đằng trước.

# 3. PHƯƠNG THỨC PHÁ HỦY

- Ví dụ minh họa: Hãy khai báo và định nghĩa phương thức phá hủy cho lớp đối tượng CPhanSo.

- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        // Nhóm phương thức và biến
18.        ~CPhanSo ();
19.} ;
    
```

Không có giá trị  
trả về.

# 3. PHƯƠNG THỨC PHÁ HỦY

- Ví dụ minh họa: Hãy khai báo và định nghĩa phương thức phá hủy cho lớp đối tượng CPhanSo.

- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        // Nhóm phương thức và biến
18.        ~CPhanSo ();
19.} ;
    
```

Không có tham  
số đầu vào.



# 3. PHƯƠNG THỨC PHÁ HỦY

- Ví dụ minh họa: Hãy khai báo và định nghĩa phương thức phá hủy cho lớp đối tượng CPhanSo.

- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        // Nhóm phương thức xử lý
18.        ~CPhanSo ();
19.} ;
    
```

Phương thức phá hủy thuộc nhóm các phương thức xử lý.

# 3. PHƯƠNG THỨC PHÁ HỦY

- Ví dụ minh họa: Hãy khai báo và định nghĩa phương thức phá hủy cho lớp đối tượng CPhanSo.

- Khai báo lớp.

```

11.class CPhanSo
12.{
13.    private:
14.        int tu;
15.        int mau;
16.    public:
17.        // Nhóm phương thức phá hủy
18.        ~CPhanSo ();
19.} ;
    
```

Có và chỉ có duy  
nhất một phương  
thức phá hủy  
trong 1 lớp mà  
thôi.

# 3. PHƯƠNG THỨC PHÁ HỦY

- Đặc điểm của phương thức phá hủy:
  - + Tên phương thức trùng với tên lớp nhưng có dấu ngã ở đằng trước.
  - + Không có giá trị trả về.
  - + Không có tham số đầu vào.
  - + Được tự động gọi thực hiện khi đối tượng hết phạm vi sử dụng.
  - + Phương thức phá hủy thuộc nhóm các phương thức xử lý.
  - + Có và chỉ có duy nhất một phương thức phá hủy trong 1 lớp mà thôi.
  - + Trong một quá trình sống của đối tượng có và chỉ có một lần phương thức phá hủy được gọi thực hiện mà thôi.

# 3. PHƯƠNG THỨC PHÁ HỦY

— Định nghĩa phương thức phá hủy.

```
1. CPhanSo::~~CPhanSo()
```

```
2. {
```

```
3. |     return;
```

```
4. }
```

# 3. PHƯƠNG THỨC PHÁ HỦY

- **Hướng dẫn sử dụng:**  
Hãy cho biết đoạn chương trình sau có bao nhiêu phương thức được gọi thực hiện. Biết rằng trong lớp đối tượng **CPhanSo** ta đã định nghĩa 3 phương thức thiết lập cơ bản (**constructors**) và phương thức phá hủy (**destructor**).

- Đoạn chương trình:

```

11.int x;
12.int y;
13.x = 5;
14.y = 7;
15.if (y>x)
16.{
17.    CPhanSo a;
18.    a.Nhap();
19.    a.Xuat();
20.}
21.cout<<x<<y;

```

# 3. PHƯƠNG THỨC PHÁ HỦY

## — Kết quả:

- + Phạm vi hoạt động của đối tượng a bắt đầu từ dòng 17 và kết thúc tại dòng 20.
- + Bởi vì đối tượng a được khai báo trong **khối lệnh** (*block*) bắt đầu từ dòng 16 và kết thúc ở dòng 20.
- + Có 4 phương thức được gọi thực hiện.

## — Đoạn chương trình:

```

11.int x;
12.int y;
13.x = 5;
14.y = 7;
15.if (y>x)
16.{
17.    CPhanSo a;
18.    a.Nhap();
19.    a.Xuat();
20.}
21.cout<<x<<y;
  
```

# 3. PHƯƠNG THỨC PHÁ HỦY

– Có 4 phương thức được gọi thực hiện.

- + Đối tượng a gọi thực hiện phương thức thiết lập mặc định (dòng 17).
- + Đối tượng a gọi thực hiện phương thức nhập (dòng 18).
- + Đối tượng a gọi thực hiện phương thức xuất (dòng 19).
- + Đối tượng a gọi thực hiện phương thức phá hủy (dòng 20).

– Đoạn chương trình:

```

11.int x;
12.int y;
13.x = 5;
14.y = 7;
15.if (y>x)
16.{
17.    CPhanSo a;
18.    a.Nhap();
19.    a.Xuat();
20.}
21.cout<<x<<y;
  
```

# 4. BÀI TẬP

— Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản và phương thức phá hủy 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**).



# 4. BÀI TẬP

— Hãy khai báo và định nghĩa các phương thức thiết lập cơ bản và phương thức phá hủy 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.