UIT Trường Đại học Công nghệ Thông tin | Khoa Khoa học và Kỹ thuật Thông tin

BAN HỌC TẬP

BUỔI 1

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

TỔNG QUAN – ĐỊNH NGHĨA

CƠ BẢN VỀ OOP
TRONG C++

TRAINER

Lê Phi Long
KTPM2020

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. Cặp hàm get/set
6. Con trỏ this
7. Thành phần static
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. Friend

Nội dung tìm hiểu:

1. Các phương pháp lập trình

2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. Cặp hàm get/set
6. Con trỏ this
7. Thành phần static
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. Friend

Các phương pháp lập trình

1. Lập trình phi cấu trúc
2. Lập trình có cấu trúc
3. Lập trình hướng đối tượng

Các phương pháp lập trình

1. Lập trình phi cấu trúc

Sử dụng trong các ngôn ngữ lập trình bậc thấp như Assembly, MIPS, …

Lập trình phi cấu trúc chỉ là lập trình để giải quyết vấn đề. Nó không tạo ra một cấu trúc logic

Các phương pháp lập trình

2. Lập trình hướng cấu trúc

Sử dụng ở một số ngôn ngữ lập trình như C, Pascal,…

Chương trình được chia thành các hàm thực hiện các chức năng khác nhau

Chương trình = Cấu trúc dữ liệu + Giải thuật

Các phương pháp lập trình

3. Lập trình hướng đối tượng

Sử dụng ở một số ngôn ngữ lập trình như C#, Java, …

Là phương pháp lập trình lấy đối tượng làm nền tảng để xây dựng chương trình

Một chương trình hướng đối tượng sẽ bao gồm các tập đối tượng có quan hệ với nhau (xem ở chương 6)

4 đặc tính cơ bản của lập trình hướng đối tượng

1. Tính đóng gói (Encapsulation)
2. Tính trừu tượng (Abstraction)
3. Tính kế thừa (Inheritance)
4. Tính đa hình (Polymorphism)

4 đặc tính cơ bản của lập trình hướng đối tượng

1. **Tính đóng gói (Encapsulation)**
2. Tính trừu tượng (Abstraction)
3. Tính kế thừa (Inheritance)
4. Tính đa hình (Polymorphism)

4 đặc tính cơ bản của lập trình hướng đối tượng

1. Tính đóng gói (Encapsulation)

Tính đóng gói cho phép che giấu thông tin và những tính chất xử lý bên trong của đối tượng. Các đối tượng khác không thể tác động trực tiếp đến dữ liệu bên trong và làm thay đổi trạng thái của đối tượng mà bắt buộc phải thông qua các phương thức công khai do đối tượng đó cung cấp.

4 đặc tính cơ bản của lập trình hướng đối tượng

1. Tính đóng gói (Encapsulation)
2. **Tính trừu tượng (Abstraction)**
3. Tính kế thừa (Inheritance)
4. Tính đa hình (Polymorphism)

4 đặc tính cơ bản của lập trình hướng đối tượng

2. Tính trừu tượng (Abstraction)

Tính trừu tượng giúp loại bỏ những thứ phức tạp, không cần thiết của đối tượng và chỉ tập trung vào những gì cốt lõi, quan trọng.

4 đặc tính cơ bản của lập trình hướng đối tượng

1. Tính đóng gói (Encapsulation)
2. Tính trừu tượng (Abstraction)
3. **Tính kế thừa (Inheritance)**
4. Tính đa hình (Polymorphism)

4 đặc tính cơ bản của lập trình hướng đối tượng

3. Tính kế thừa (Inheritance)

Tính kế thừa cho phép xây dựng một lớp mới (lớp Con), kế thừa và tái sử dụng các thuộc tính, phương thức dựa trên lớp cũ (lớp Cha) đã có trước đó.

4 đặc tính cơ bản của lập trình hướng đối tượng

1. Tính đóng gói (Encapsulation)
2. Tính trừu tượng (Abstraction)
3. Tính kế thừa (Inheritance)
4. **Tính đa hình (Polymorphism)**

4 đặc tính cơ bản của lập trình hướng đối tượng

4. Tính đa hình (Polymorphism)

Tính đa hình cho phép các đối tượng khác nhau thực thi chức năng giống nhau theo những cách khác nhau.

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. **Lớp và đối tượng**
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. Cặp hàm get/set
6. Con trỏ this
7. Thành phần static
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. Friend

Lớp và đối tượng

1. Lớp đối tượng là gì?

Các đối tượng có các đặc tính tương tự nhau được gom chung thành lớp đối tượng, là một mô tả trừu tượng của các nhóm đối tượng cùng bản chất

Lớp trong C++ thực chất là một kiểu dữ liệu do người dùng định nghĩa

Cú pháp khai báo lớp trong C++:

class <Tên lớp> {...};

Lớp và đối tượng

2. Đối tượng là gì?

Đối tượng là một thể hiện cụ thể của một lớp

Trong C++, đối tượng thuộc một lớp thực chất là một biến mang kiểu dữ liệu lớp đó

Cách khai báo đối tượng thuộc lớp:

<Tên lớp> <Tên đối tượng>;

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. **Thuộc tính và phương thức**
4. Phạm vi truy cập
5. Cặp hàm get/set
6. Con trỏ this
7. Thành phần static
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. Friend

Thuộc tính và phương thức

Một lớp đối tượng được đặc trưng bởi các thuộc tính và các thao tác của các đối tượng thuộc lớp

Thuộc tính: Là thành phần của đối tượng, có giá trị nhất định cho mỗi đối tượng trong hệ thống

Thao tác: Thể hiện hành vi của một đối tượng tác động qua lại với các đối tượng khác hoặc với chính nó

Mỗi thao tác trên một lớp ứng với một cài đặt cụ thể. Một cài đặt như vậy gọi là một phương thức

Thuộc tính và phương thức

Các thuộc tính được khai báo như là khai báo biến. Nói cách khác trong khi khai báo lớp, các biến dữ liệu thuộc lớp sẽ được gọi là các thuộc tính

Các phương thức được khai báo như là khai báo hàm. Nói cách khác khi khai báo hàm, các hàm thực thi thuộc lớp sẽ được gọi là các phương thức

Cách gọi thuộc tính/phương thức của đối tượng thuộc lớp:

<tên đối tượng>.<tên thuộc tính/phương thức>…;

<tên con trỏ đối tượng>-><tên thuộc tính/phương thức>…;

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. **Phạm vi truy cập**
5. Cặp hàm get/set
6. Con trỏ this
7. Thành phần static
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. Friend

Phạm vi truy cập

private: Chỉ các thành phần được khai báo private chỉ được truy xuất trong nội bộ lớp

public: Các thành phần được khai báo public được truy xuất ở mọi nơi trong chương trình

protected: ???

Phạm vi truy cập mặc định trong C++ là private

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. **Cặp hàm get/set**
6. Con trỏ this
7. Thành phần static
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. Friend

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. Cặp hàm get/set
6. **Con trỏ this**
7. Thành phần static
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. Friend

Con trỏ this

Từ khóa this trong định nghĩa của các hàm thành phần lớp dung để xác định địa chỉ của đối tượng dùng làm tham số ngầm định cho hàm thành phần

Con trỏ this tham chiếu đến đối tượng đang gọi hàm thành phần

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. Cặp hàm get/set
6. Con trỏ this
7. **Thành phần static**
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. Friend

Thành phần static

Thành phần tatic trong C++ là dữ liệu của lớp không phải là dữ liệu của đối tượng.

Thành phần static xuất hiện trước khi khởi tạo đối tượng của lớp, và nó chỉ tồn tại duy nhất.

Một hàm khai báo là static không được phép truy cập những thành phần non-static của lớp.

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. Cặp hàm get/set
6. Con trỏ this
7. Thành phần static
8. **Phương thức khởi tạo**
9. Phương thức phá hủy
10. Friend

Phương thức khởi tạo

Phương thức khởi tạo là một phương thức đặc biệt được tự động gọi khi một đối tượng được khởi tạo

Phương thức khởi tạo thường được dùng để xây dựng những khởi tạo cần thiết khi một đối tượng được tạo lập

Phương thức khởi tạo là một phương thức không có kiểu dữ liệu trả về, tên của phương thức trùng với tên của lớp

Cú pháp khai báo: <Tên lớp>(Danh sách tham số nếu có);

Trình biên dịch sẽ tự động trang bị một phương thức khởi tạo mặc định nếu người dùng không khai báo bất kì một phương thức khởi tạo nào khác

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. Cặp hàm get/set
6. Con trỏ this
7. Thành phần static
8. Phương thức khởi tạo
9. **Phương thức phá hủy**
10. Friend

Phương thức phá hủy

Phương thức phá hủy là một phương thức đặc biệt được tự động gọi khi một đối tượng không còn được sử dụng

Phương thức phá hủy thường được dùng để dọn dẹp cần thiết khi một đối tượng không còn được sử dụng

Phương thức phá hủy là một phương thức không có kiểu dữ liệu trả về, tên của phương thức trùng với tên của lớp, không có tham số truyền vào. Mỗi lớp có duy nhất một phương thức phá hủy

Cú pháp khai báo: ~<Tên lớp>();

Nội dung tìm hiểu:

1. Các phương pháp lập trình
2. Lớp và đối tượng
3. Thuộc tính và phương thức
4. Phạm vi truy cập
5. Cặp hàm get/set
6. Con trỏ this
7. Thành phần static
8. Phương thức khởi tạo
9. Phương thức phá hủy
10. **Friend**

Thành phần friend

Thành phần friend là một thành phần không thuộc lớp nhưng nó được quyền truy xuất tất cả các thành phần private của lớp

Hãy lưu ý khi sử dụng hàm friend vì nó sẽ phá hủy tính đóng gói và che giấu dữ liệu trong lập trình hướng đối tượng

UIT Trường Đại học Công nghệ Thông tin | Khoa Khoa học và Kỹ thuật Thông tin

BAN HỌC TẬP

BUỔI 1

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

OVERLOADING HÀM VÀ TOÁN TỬ

Operator và Overload operator trong C++

TRAINER

Lê Phi Long
KTPM2020

Nội dung tìm hiểu:

1. Overload là gì?
2. Operator là gì?
3. Sự cần thiết phải overload toán tử
4. Một số cú pháp overload toán tử cần phải nhớ

Nội dung tìm hiểu:

1. **Overload là gì?**
2. Operator là gì?
3. Sự cần thiết phải overload toán tử
4. Một số cú pháp overload toán tử cần phải nhớ

Overload là gì?

Overload (nạp chồng): là việc tạo ra nhiều phương thức có cùng tên thực hiện chức năng tương tự nhau nhưng khác nhau về tham số đầu vào (kiểu dữ liệu hoặc số lượng tham số đầu vào)

Overload là một yếu tố tạo nên tính đa hình trong OOP

Overload thực hiện đa hình trong compile time

Nội dung tìm hiểu:

1. Overload là gì?
2. **Operator là gì?**
3. Sự cần thiết phải overload toán tử
4. Một số cú pháp overload toán tử cần phải nhớ

Operator

Các toán tử cho phép ta sử dụng cú pháp toán học đối với các kiểu dữ liệu của C++ thay vì gọi hàm

Có các loại toán tử một ngôi như ++, --, &, *, … và toán tử hai ngôi như +, -, *, /, …

Sự cài đặt phép toán chỉ cho phép tạo ra phép toán mới trên cơ sở kí hiệu phép toan đã có, không được quyền cài đặt các phép toán mới

Một toán tử có thể dùng cho nhiều kiểu dữ liệu

Nội dung tìm hiểu:

1. Overload là gì?
2. Operator là gì?
3. **Sự cần thiết phải overload toán tử**
4. Một số cú pháp overload toán tử cần phải nhớ

Lợi ích khi Overload toán tử

Xét bài toán: Nhập vào các phân số a, b, c, d và tiến hành tính giá trị biểu thức sau:

(a+b*c)/(d-a)+(b*d-2*a)/(3*c)

Lợi ích khi Overload toán tử

Viết theo cách thông thường: (a+b*c)/(d-a+(b*d-2*a))

```
PhanSo a, b, c, d;
PhanSo e(2);
a.Cong(b.Nhan(c)).Chia(d.Tru(a).Cong(b.Nhan(d).Tru(e.Nhan(a)))).Xuat();
```

Lợi ích khi Overload toán tử

Gần với kiểu trình bày mà con người quen dùng

Đơn giản hóa mã chương trình

Cú pháp chung để overload toán tử:

<Tên kiểu dữ liệu trả về> operator<Tên operator>(Danh sách tham số nếu có);

Nội dung tìm hiểu:

1. Overload là gì?
2. Operator là gì?
3. Sự cần thiết phải overload toán tử
4. **Một số cú pháp overload toán tử cần phải nhớ**

Các toán tử 2 ngôi

Thường đi kèm friend và một bộ constructor để thực hiện phép chuyển kiểu

Ví dụ: Các phép toán +, -, *, /:

PhanSo(int Tu = 0, int Mau = 0);

friend PhanSo operator+(PhanSo a, PhanSo b);

Khi đó các phép toan PhanSo + PhanSo, PhanSo + int và int + PhanSo sẽ được thực hiện mà không cần cài đặt cả ba phiên bản

Toán tử nhập – xuất

Cú pháp:

```
friend istream& operator>>(istream& is, PhanSo& a)
{
        //Thực hiện việc nhập
        return is;
}
friend ostream& operator<<(ostream& os, PhanSo a)
{
        //Thực hiện việc xuất
        return os;
}
```

Toán tử ++, --

Cú pháp:

PhanSo& operator++(); //Dành cho ++a;

PhanSo operator++(int); //Dành cho a++;

Phép toán lấy phần tử mảng

Ta có thể định nghĩa phép toán [ ] để truy xuất phần tử của một đối tượng có ý nghĩa mảng. Ví dụ DanhSachSinhVien

```
class DanhSachSinhVien {
        int n;
        SinhVien* List;
public:
        SinhVien& operator[](int i) { return List[i]; }
        SinhVien operator[] (int i) const { return List[i]; } //Dành cho đối tượng gọi hàm là một hằng


};
```

UIT Trường Đại học Công nghệ Thông tin | Khoa Khoa học và Kỹ thuật Thông tin

BAN HỌC TẬP

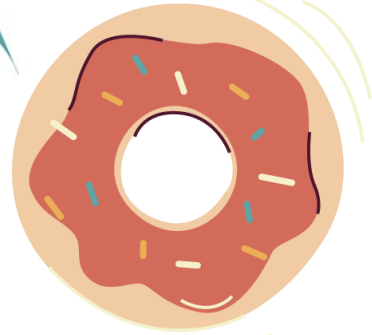
BUỔI 1
LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

XIN CHÂN THÀNH CẢM ƠN

Chúc các bạn thi tốt
có nhiều học bổng nha

TRAINER

Lê Phi Long
KTPM2020



LINK ĐIỂM DANH

https://bit.ly/Buoi1_OOP