Phương pháp lập trình hướng đối tượng

Tuần 06:

Kế thừa

Phạm Tú San
ptsan@fit.hcmus.edu.vn

Nội dung

- **Khái niệm kế thừa.**
- Tầm vực trong kế thừa.
- Định nghĩa lại phương thức.
- Quan hệ IS-A và HAS-A.
- Bài tập.

Khái niệm kế thừa

- Vấn đề trùng lắp thông tin:
  - Nhiều lớp có thông tin giống nhau.
  - Có 2 dạng:
    - Dạng chia sẻ: $A \cap B \neq \emptyset$.
    - Dạng mở rộng: $B = A + \varepsilon$.
  - Nhược điểm:
    - Xây dựng tốn kém.
    - Dung lượng lưu trữ lớn.
    - Thay đổi phần chung khó khăn.

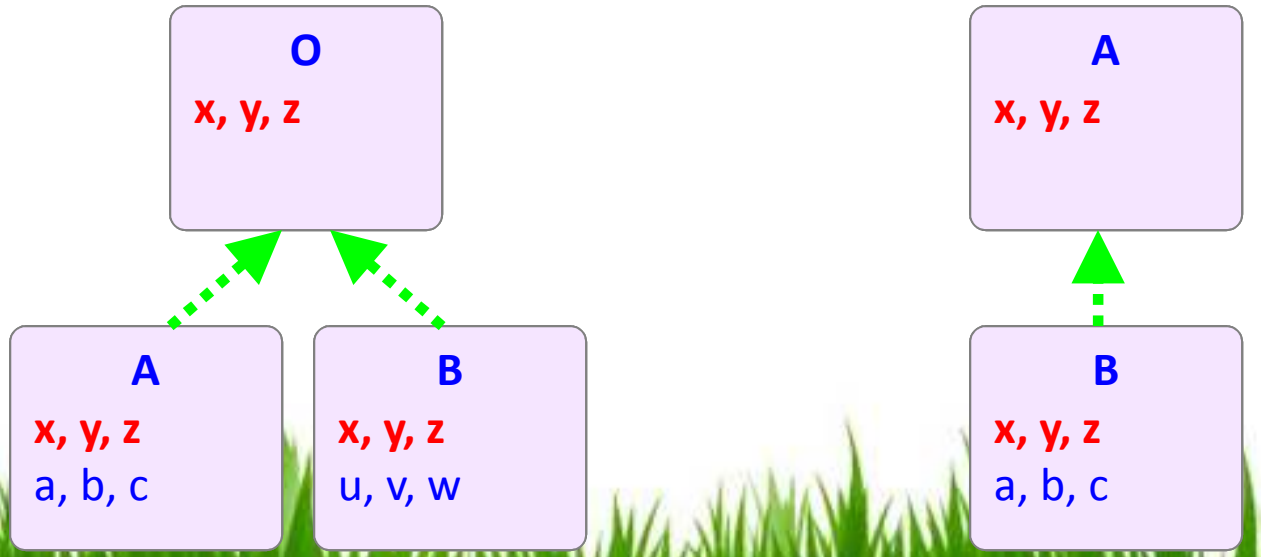
A	B
x, y, z a, b, c	x, y, z u, v, w

A	B
x, y, z	x, y, z a, b, c

Giải quyết: tái sử dụng!!

Khái niệm kế thừa

- Khái niệm kế thừa:
  - Định nghĩa lớp mới dựa trên những lớp đã có.
  - **Lớp cơ sở:** lớp dùng để định nghĩa lớp mới.
  - **Lớp kế thừa:** lớp được định nghĩa từ lớp đã có.
  - Lớp kế thừa thừa hưởng **TẤT CẢ** từ lớp cơ sở.



Ví dụ kế thừa

Giáo viên

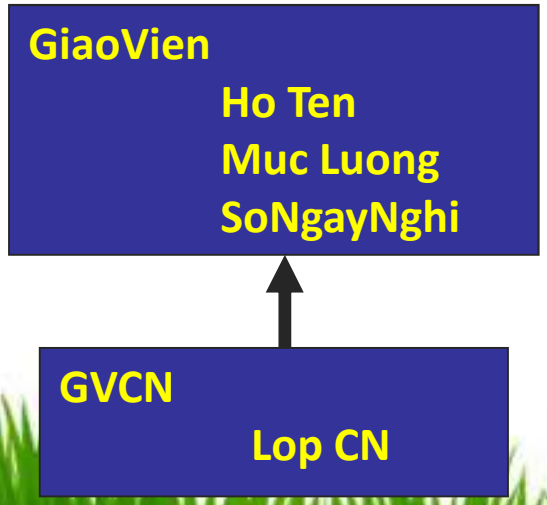
Thông tin:
- Họ tên.
- Mức lương.
- Số ngày nghỉ.

Công việc:
- Giảng dạy.
- Tính lương.

Thông tin:
- Họ tên.
- Mức lương.
- Số ngày nghỉ.
- **Lớp chủ nhiệm.**

Công việc:
- Giảng dạy.
- Tính lương.
- **Sinh hoạt chủ nhiệm.**

GVCN



Khái niệm kế thừa

- Khai báo trong C++:

  class <Lớp kế thừa> : <Loại kế thừa> <Lớp cơ sở>

- Loại kế thừa:
  - public, private, protected.
- Ví dụ:

```
class A : public O
{
private:
        // Khai báo thuộc tính mới của A.
public:
        // Khai báo phương thức mới của A.
};
```

Khái niệm kế thừa

- Ví dụ:

Lớp cơ sở

```
class GiaoVien
{
private:
        string    mHoTen;
        float     mMucLuong;
        int       mSoNgayNghi;
public:
        GiaoVien(string HoTen,
                 float fMucLuong,
                 int iSoNgayNghi);
        void giangDay();
        float tinhLuong();
};
```

Lớp kế thừa

```
class GVCN : public GiaoVien
{
private:
    string   mLopCN;
public:
    GVCN(string HoTen,
         float fMucLuong,
         int iSoNgayNghi,
         string sLopCN);
    void sinhHoatCN();
};
```

GVCN thừa hưởng TẤT CẢ thuộc tính và phương thức của GiaoVien

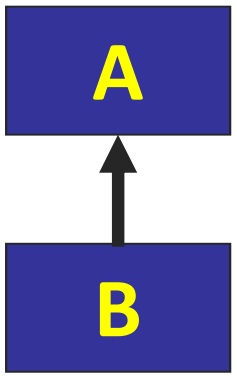
Khái niệm kế thừa

- Ví dụ:

```
void main()
{
        GiaoVien   gv1("Minh", 500000, 5);
        gv1.giangDay();
        float fLuong1 = gv1.tinhLuong();

        GVCN   gv2("Hanh", 700000, 3, "11A");
        gv2.giangDay();
        gv2.sinhHoatCN();
        float fLuong2 = gv2.tinhLuong();
}
```

Một số lưu ý trong kế thừa



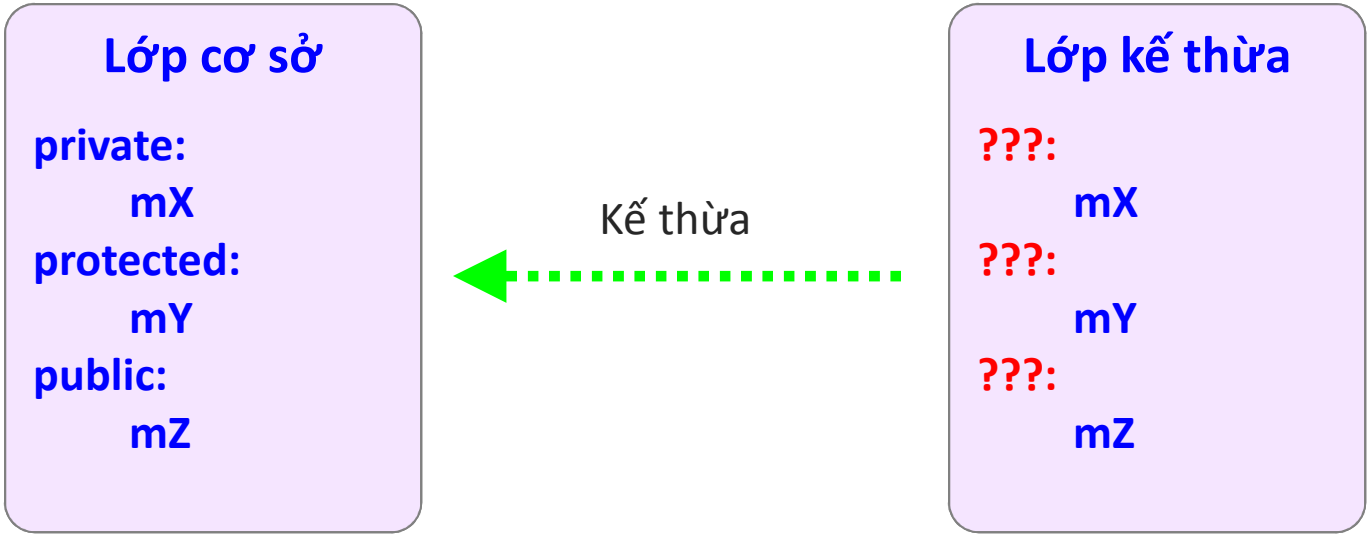
- Các thành phần thuộc tính và hành động **`public`** của A sẽ là các thành phần trong B
- Các thành phần **`private`** của A sẽ là 1 phần trong B nhưng chỉ được truy xuất qua các hàm **`public`** hay **`protected`** của A

Nội dung

- Khái niệm kế thừa.
- **Tầm vực trong kế thừa.**
- Định nghĩa lại phương thức.
- Quan hệ IS-A và HAS-A.
- Bài tập.

Tầm vực trong kế thừa

- **Tầm vực thay đổi thế nào khi kế thừa?**



➔ **Loại kế thừa quyết định!!**

Tầm vực trong kế thừa

- Bảng tầm vực trong kế thừa:

Tầm vực	Kế thừa public	Kế thừa protected	Kế thừa private
public	public	protected	private
protected	protected	protected	private
private	***Không thể truy xuất***	***Không thể truy xuất***	***Không thể truy xuất***

```
class A
{
private:
        int mX;
protected:
        int mY;
public:
        int mZ;
};
```

```
class B: private A
{
};
```

```
void main()
{
        B b;
        cout<<b.mZ;
        cout<<b.mY;
        cout<<b.mX;
}
```

Nội dung

- Khái niệm kế thừa.
- Tầm vực trong kế thừa.
- **Định nghĩa lại phương thức.**
- Quan hệ IS-A và HAS-A.
- Bài tập.

Định nghĩa lại phương thức

- Kế thừa một phần:
  - Không kế thừa "máy móc" tất cả.
  - **Lớp kế thừa có thể thay đổi những gì đã kế thừa!!**

    ➔ Định nghĩa lại phương thức đã kế thừa.

Định nghĩa lại phương thức

- Ví dụ:
  - GVCN kế thừa từ GiaoVien.
  - GVCN tính lương khác GiaoVien.
    - Lương GV = Mức lương – Số ngày nghỉ * 10000.
    - Lương GVCN = Lương GV + Phụ cấp 50000.

    ➔ **Viết lại phương thức tinhLuong() cho lớp GVCN.**

Ví dụ - định nghĩa lại phương thức

```
class GiaoVien
{
private:
        string   mHoTen;
        float    mMucLuong;
        int      mSoNgayNghi;
public:
        GiaoVien(string HoTen, float fMucLuong, int iSoNgayNghi);
        void giangDay();
        float tinhLuong()
        {
                return mMucLuong – mSoNgayNghi * 10000;
        }
};
```

Ví dụ - định nghĩa lại phương thức (tt)

```
class GVCN : public GiaoVien
{
private:
        string   mLopCN;
public:
        GVCN(string sTen, float fLuong,
                int iNgayNghi,
                string  sLopCN);
        void sinhHoatCN();
        float tinhLuong()
        {
        return GiaoVien::TinhLuong()
        + 50000;
        }
};
```

```
void main()
{
    GiaoVien   gv1("Minh", 500000, 5);
    gv1.giangDay();
    float fLuong1 = gv1.tinhLuong();

    GVCN   gv2("Hanh", 700000, 3);
    gv2.giangDay();
    float fLuong2 = gv2.tinhLuong();
}
```

Nội dung

- Khái niệm kế thừa.
- Tầm vực trong kế thừa.
- Định nghĩa lại phương thức.
- **Quan hệ IS-A và HAS-A.**
- Bài tập.

Quan hệ IS-A và HAS-A

- Quan hệ IS-A:
  - Lớp A quan hệ IS-A với lớp B
    - A là một trường hợp đặc biệt của B.
    - A cùng loại với B.
- Ví dụ:
  - GVCN là một GiaoVien đặc biệt.
  - HinhVuong là một HinhChuNhat đặc biệt.
  - ConMeo là một ConVat đặc biệt.

Quan hệ IS-A và HAS-A

- Quan hệ HAS-A:
  - Lớp A quan hệ HAS-A với lớp B
    - A bao hàm B.
    - A chứa B.
    - B là một bộ phận của A.
- Ví dụ:
  - ChiecXe chứa BanhXe.
  - QuyenSach chứa TrangSach.

Quan hệ IS-A và HAS-A

- **Luật xây dựng lớp.**
  - A có quan hệ IS-A với B.

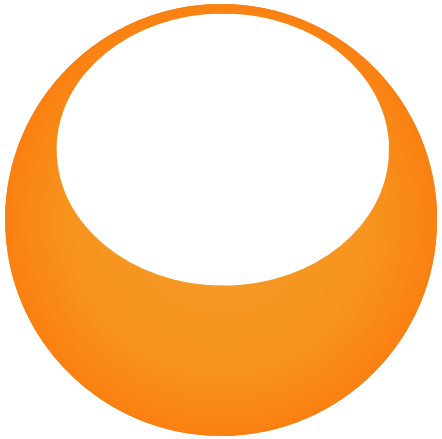
    ➔ **Cho A kế thừa B.**
  - A có quan hệ HAS-A với B.

    ➔ **Cho B là một thuộc tính của A.**
- Ví dụ:

```
class ConMeo : public ConVat { };
class ChiecXe
{
private:
        BanhXe   *mBanhXe;
};
```

Tóm tắt

- Khái niệm kế thừa:
  - Định nghĩa lớp mới dựa trên những lớp đã có.
  - Lớp kế thừa thừa hưởng tất cả từ lớp cơ sở.
- Tầm vực trong kế thừa:
  - Tầm vực thay đổi tùy theo loại kế thừa.
- Định nghĩa lại phương thức:
  - Thay đổi những phương thức kế thừa từ lớp cơ sở.
- Quan hệ IS-A và HAS-A:
  - IS-A: A là trường hợp đặc biệt của B => A kế thừa B.
  - HAS-A: A bao hàm B => B là thuộc tính của A.

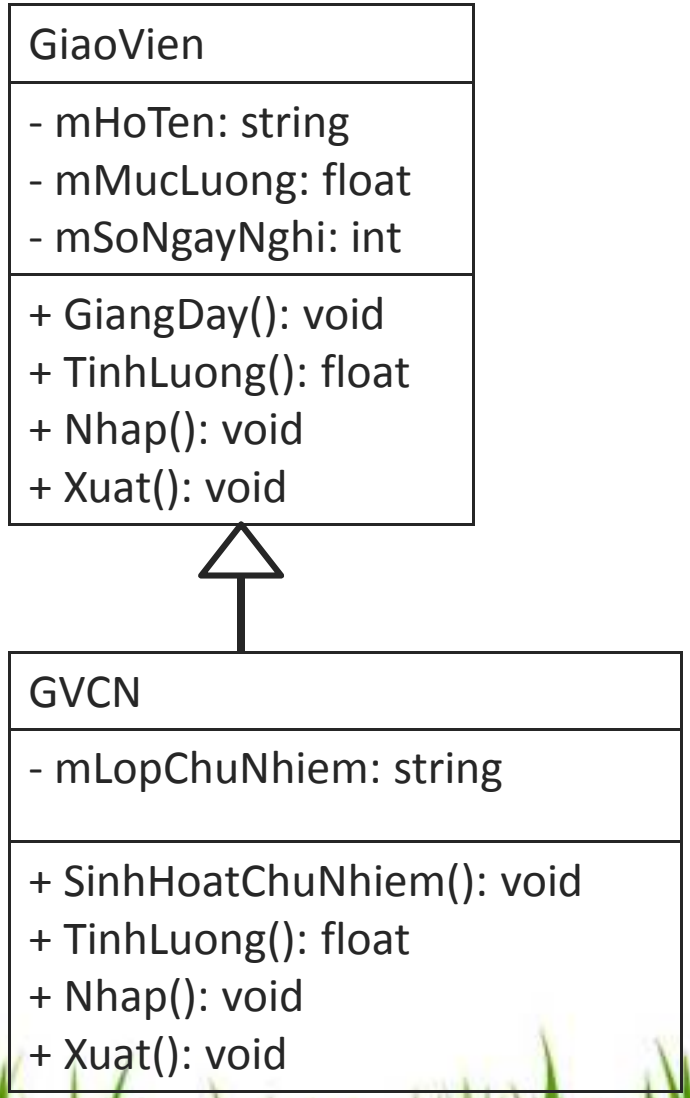


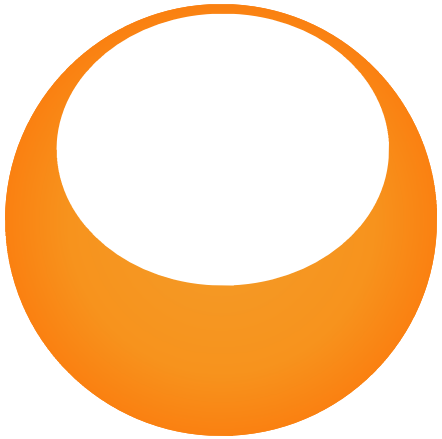
SƠ ĐỒ LỚP

Sơ đồ lớp – quan hệ kế thừa và chứa trong

```
class ConMeo : public ConVat { };
class ChiecXe
{
private:
        BanhXe   *mBanhXe;
};
```

Sơ đồ lớp mức chi tiết





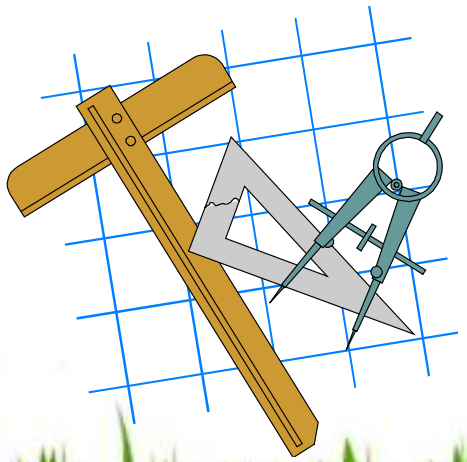
BÀI TẬP

Bài tập 9.1

Những cặp đối tượng sau có quan hệ IS-A hay HAS-A?

Vẽ sơ đồ lớp sau đó khai báo lớp cho từng cặp thể hiện quan hệ giữa chúng.

- Hình vuông / Hình chữ nhật.
- Đa giác / Cạnh.
- Giám đốc / Nhân viên.
- Hình tròn / Hình Ellipse.
- Máy bay / Động cơ.
- Câu / Từ.
- Mỹ phẩm / Hàng hóa.
- Cây lúa / Cây lương thực.
- Thư viện / Sách.
- Phim hoạt hình / Phim ảnh.



Bài tập 9.2

Cho lớp TaiKhoan:

```
class TaiKhoan
{
private:
        float       mSoDu = 0;
public:
        float baoSoDu() {   return mSoDu;   }
        void napTien(float fSoTien) {   mSoDu += fSoTien;   }
        void rutTien(float fSoTien)
        {
                if (fSoTien <= mSoDu)
                        mSoDu -= fSoTien;
        }
};
```

Bài tập 9.2 (tt)

Dựa trên lớp TaiKhoan, xây dựng lớp TaiKhoanTietKiem như sau:

- Có thêm thông tin:
  - Kỳ hạn gửi.
  - Lãi suất.
  - Số tháng đã gửi.
- Khi nạp tiền, số tháng đã gửi được tính lại từ đầu.
- Chỉ được rút tiền khi đến kỳ hạn.
- Cho phép tăng số tháng đã gửi.
- Tính số dư tại thời điểm hiện tại.

Bài tập 9.3

Một chiếc xe máy chạy 100km tốn 2lit xăng, cứ chở thêm 10kg hàng xe tốn thêm 0.1lit xăng.

Một chiếc xe tải chạy 100km tốn 20lit xăng, cứ chở thêm 1000kg hàng xe tốn thêm 1lit xăng.

Dùng kế thừa xây dựng lớp XeMay và XeTai cho phép:

- Chất một lượng hàng lên xe.
- Bỏ bớt một lượng hàng xuống xe.
- Đổ một lượng xăng vào xe.
- Cho xe chạy một đoạn đường.
- Kiểm tra xem xe đã hết xăng chưa.
- Cho biết lượng xăng còn trong xe.

Thao khảo

- Slide PPLTHDT của
  - Thầy Nguyễn Minh Huy
  - Thầy Hồ Tuấn Thanh