

JAVA OOP

Kế Thừa - Đa hình



Tại sao cần kế thừa

Nhu cầu & giới thiệu



Làm việc với kế thừa & đa hình

Code trên dự án



Luyện tập các dự án nâng cao

Phân tích nghiệp vụ & code



Inheritance



INHERITANCE

- ✓ **Inheritance** là một khái niệm rất then chốt trong OOP.
- ✓ **Inheritance** cho phép tạo một class mới dựa trên một class có sẵn.
- ✓ **Inheritance** có thể làm đơn giản hóa thiết kế và cài đặt của ứng dụng.
- ✓ Class mới kế thừa từ class có sẵn được gọi là một **derived class**, **child class** hoặc **subclass**.
- ✓ Class có sẵn được gọi là một **base class**, **parent class** hoặc **superclass**.
- ✓ **Subclass** thừa hưởng được từ superclass các **field** và các **method** của **superclass**.
- ✓ **Subclass** có thể **extend superclass** bằng cách **bổ sung thêm** các **field**, các **constructor** và các **method**.
- ✓ **Subclass** cũng có khả năng **override** lại các **method** thừa kế của **superclass** để tạo thành **version phù hợp**.



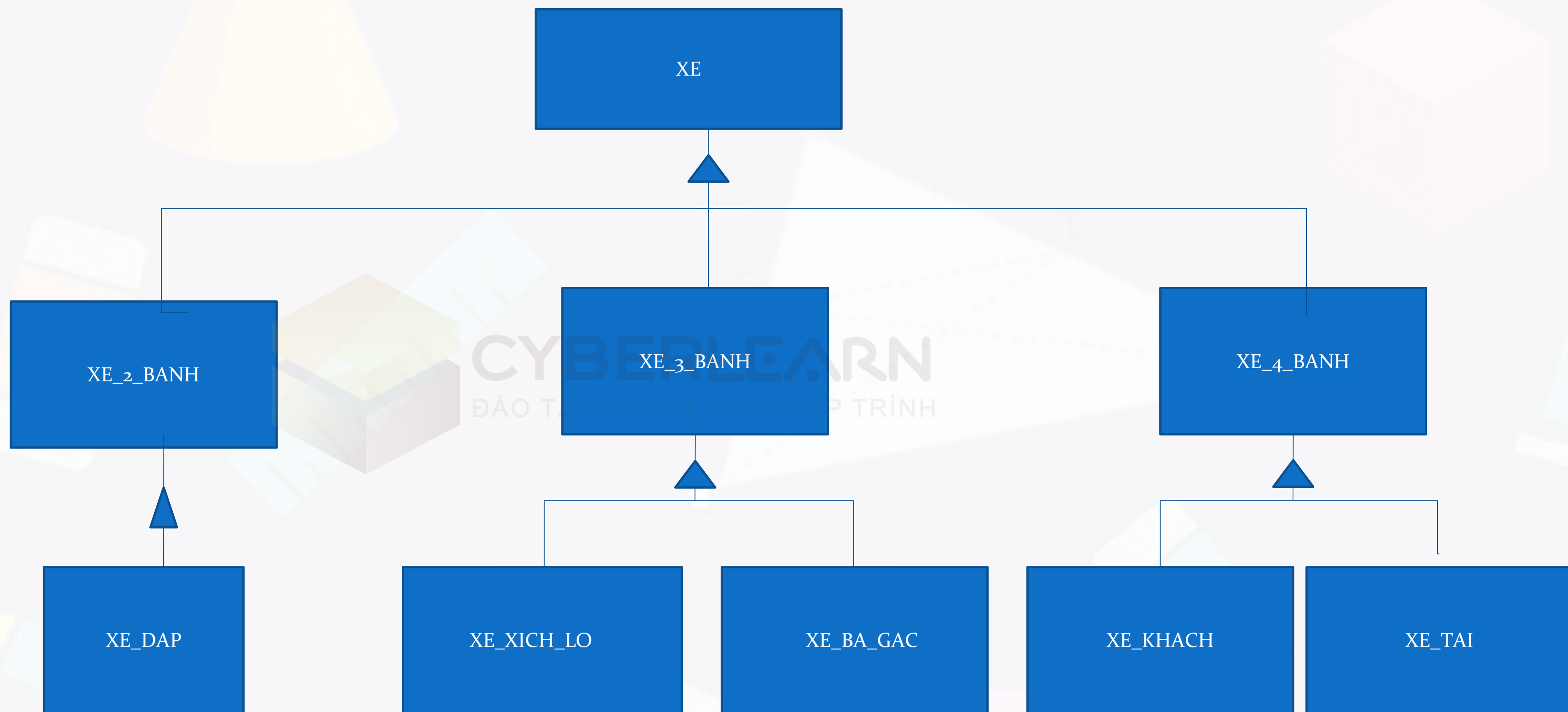
PHÂN TÍCH CÂY KẾ THỪA

- Cây kế thừa là một cây đa nhánh thể hiện mối quan hệ giữa các lớp trong hệ thống, chương trình.
- Cây kế thừa cho các lớp đối tượng:
- Lớp XE
- Lớp XE_2_BANH
- Lớp XE_3_BANH
- Lớp XE_4_BANH
- Lớp XE_XICH_LO
- Lớp XE_BA_GAC
- Lớp XE_KHACH
- Lớp XE_TAI
- Lớp XE_DAP

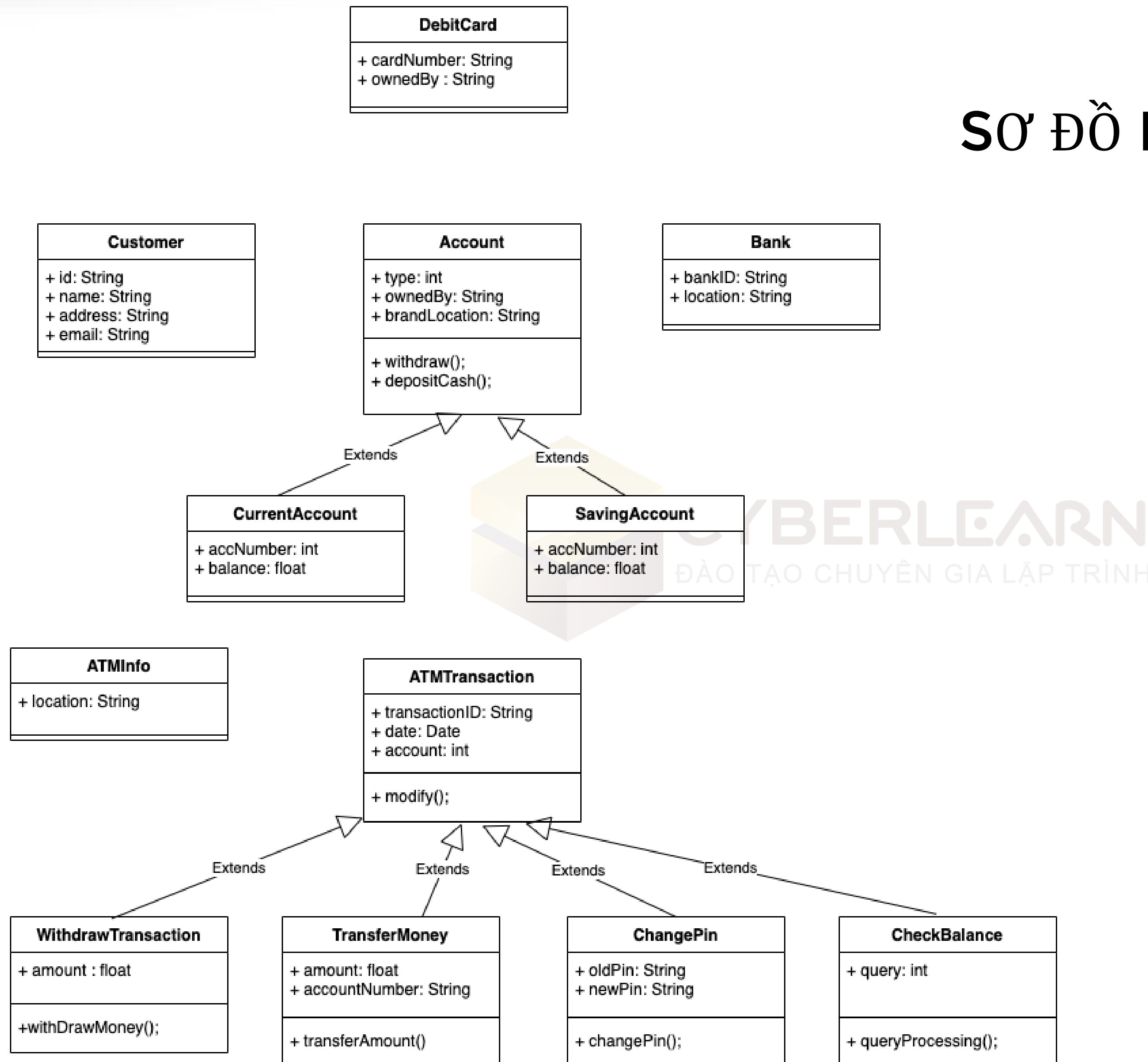


CYBERLEARN
ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

PHÂN TÍCH CÂY KẾ THỪA



SƠ ĐỒ LỚP TỔNG QUÁT



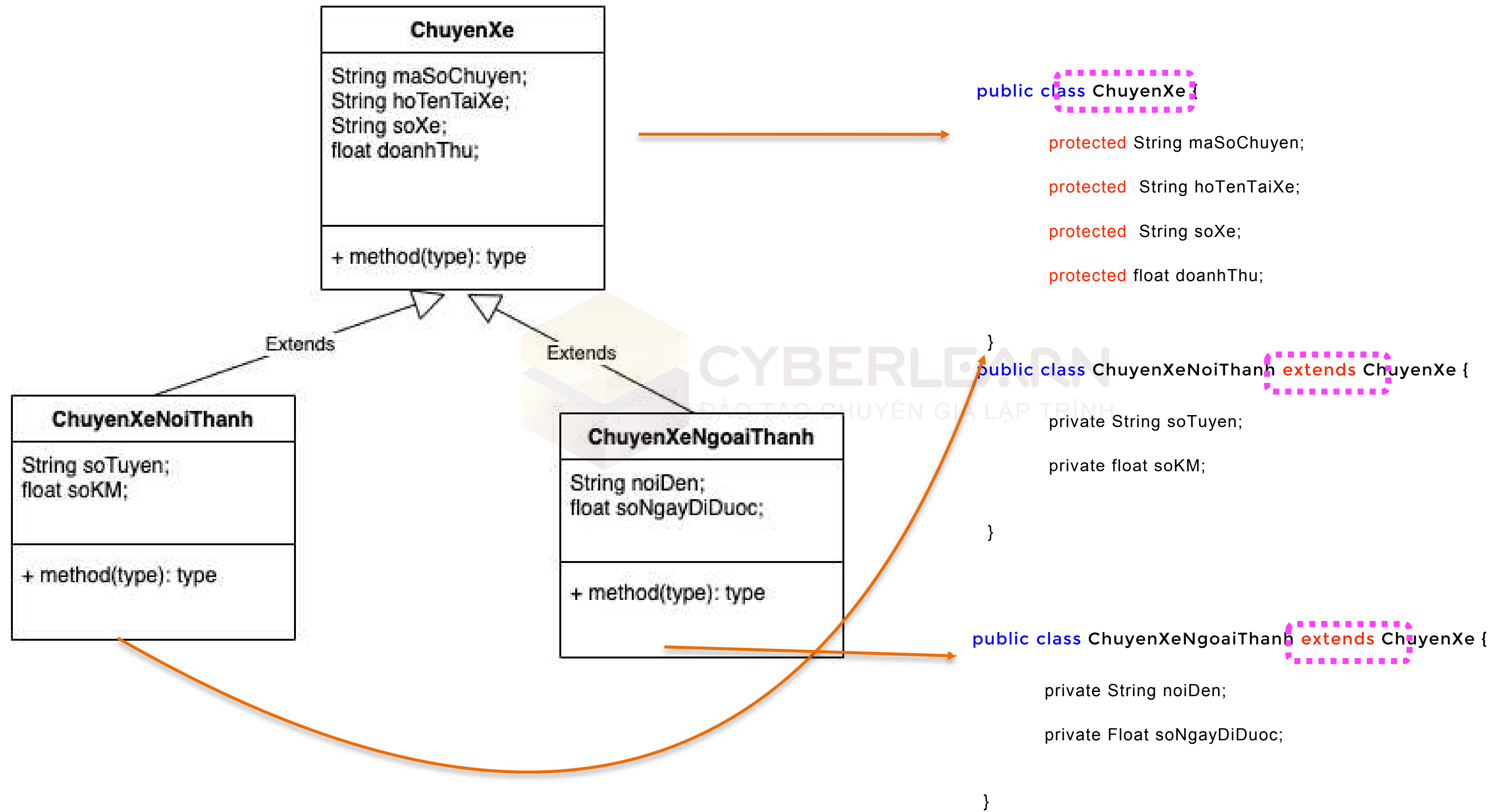
BÀI TẬP NỘP: LẬP CÂY KẾ THỪA CHO CÁC YÊU CẦU



1. Lập cây kế thừa để quản lý các đối tượng Hình ví dụ: Hình 2 chiều (Tròn, vuông, tam giác,...), Hình 3 chiều (Tứ diện, đa lăng trụ,...)
2. Lập cây kế thừa cho hệ thống quản lý nhân sự trong công ty. Ví dụ: Hội đồng quản trị, Nhân viên kinh doanh, Phòng ban, nhân viên bán thời gian,....
3. Lập cây kế thừa cho hệ thống quản lý các phân khúc và loại xe (Xe ô tô và xe máy các loại)



CODE KẾ THỪA



DẪN XUẤT (ACCESS MODIFIER)

- * **private**
 - Nhìn thấy trực tiếp trong class hiện hành.
- * **protected**
 - Nhìn thấy trực tiếp trong các class cùng package và các subclass.
- * **public**
 - Nhìn thấy trực tiếp trong các class tất cả các package.
- * **default** (no keyword)
 - Nhìn thấy trực tiếp trong các class cùng package

CONSTRUCTOR

```
public class ChuyenXe {
    protected String maSoChuyen;
    protected String hoTenTaiXe;
    protected String soXe;
    protected float doanhThu;

    public ChuyenXe(){

    }

    public ChuyenXe (String maSo, String hoTen, String
soXe, float dThu) {
        this.maSoChuyen = maSo;
        this.hoTenTaiXe = hoTen;
        this.soXe = soXe;
        this.doanhThu = dThu;
    }
}
```

Chỉ định constructor của superclass bằng từ khóa **super**

– **Vị trí** chỉ định là **dòng đầu tiên** trong cài đặt của phương thức subclass constructor.

– **super ()** : chỉ định default constructor.

– **super (...)** : chỉ định parameterized constructor.

```
public class ChuyenXeNoiThanh extends ChuyenXe {
    private String soTuyen;
    private float soKM;

    public ChuyenXeNoiThanh(){
        super (); // Chỉ định phương thức khởi tạo mặc định lớp cha
    }

    public ChuyenXeNoiThanh ( String maSo, String hoTen, String soXe, float dThu, String
soTuyen, float soKM) {
        super (maSo, hoTen, soXe, dThu); // Chỉ định phương thức khởi tạo có tham số lớp cha
        this.soTuyen = soTuyen;
        this.soKM = soKM;
    }
}
```

```
public class ChuyenXeNgoaiThanh extends ChuyenXe {
    private String noiDen;
    private float soNgayDiDuoc;

    public ChuyenXeNgoaiThanh(){
        super (); // Chỉ định phương thức khởi tạo mặc định lớp cha
    }

    public ChuyenXeNgoaiThanh ( String maSo, String hoTen, String soXe, float dThu, String noiDen, float
soNgayDi) {
        super (maSo, hoTen, soXe, dThu); // Chỉ định phương thức khởi tạo có tham số lớp cha
        this.soTuyen = noiDen;
        this.soKM = soNgayDi;
    }
}
```


JAVA OOP

Override



Lý do cần Override & cách thực hiện

Nhu cầu cần thực hiện Override & cách thực hiện



Một số ghi chú về Override

Nền tảng lý thuyết về Override



Code trên dự án

Triển khai dự án

OVERRIDE

Xét bài toán quản lý nhân sự bao gồm Nhân viên thường, Trưởng phòng và giám đốc. Mỗi nhân sự có họ lương theo ngày và phụ cấp.

Xây dựng các lớp đối tượng và tính lương cho nhân viên

```
public class NhanSu {  
    protected String hoTen;  
    protected String maSo;  
    protected String soNgayLam;  
    protected float luong;  
  
    public NhanSu(){  
  
    }  
  
    public void tinhLuong () {  
        this.luong = 0;  
    }  
}
```

```
public class NhanVienThuong extends NhanSu {  
    final int PHU_CAP = 200;  
    final int LUONG_NGAY = 100;  
  
    public NhanVienThuong(){  
  
    }  
  
    @Override  
    public void tinhLuong () {  
        this.luong = this.soNgayLam * LUONG_NGAY + PHU_CAP;  
    }  
}
```

```
public class TruongPhong extends NhanSu {  
    final int PHU_CAP = 300;  
    final int LUONG_NGAY = 200;  
  
    public TruongPhong(){  
  
    }  
  
    @Override  
    public void tinhLuong ( {  
        this.luong = this.soNgayLam * LUONG_NGAY + PHU_CAP;  
    }  
}
```


OVERRIDE

```
public class ChuyenXe {  
    protected String maSoChuyen;  
    protected String hoTenTaiXe;  
    protected String soXe;  
    protected float doanhThu;  
  
    public ChuyenXe(){  
  
    }  
}
```

```
    public void nhap (Scanner scan) {  
        System.out.println(" Nhập mã số:");  
        this.maSoChuyen = scan.nextLine();  
        System.out.println(" Nhập họ tên tài xế:");  
        this.hoTenTaiXe = scan.nextLine();  
        System.out.println(" Nhập số xe:");  
        this.soXe = scan.nextLine();  
        System.out.println(" Nhập doanh thu:");  
        this.doanhThu =  
        Float.parseFloat(scan.nextLine());  
    }  
}
```

```
public class ChuyenXeNoiThanh extends ChuyenXe {  
    private String soTuyen;  
    private float soKM;  
  
    public ChuyenXeNoiThanh(){  
        super (); // Chỉ định phương thức khởi tạo mặc định  
        lớp cha  
    }  
}
```

```
@Override  
public void nhap ( Scanner scan) {  
    super.nhap(); // Gọi phương thức nhập từ lớp cha  
    System.out.println(" Nhập số tuyến:");  
    this.soTuyen = scan.nextLine();  
    System.out.println(" Nhập số KM:");  
    this.soKM = Float.parseFloat(scan.nextLine());  
}
```

```
public class ChuyenXeNgoaiThanh extends ChuyenXe {  
    private String noiDen;  
    private float soNgayDiDuoc;  
  
    public ChuyenXeNgoaiThanh(){  
        super (); // Chỉ định phương thức khởi tạo mặc định lớp  
        cha  
    }  
  
    @Override  
    public void nhap ( Scanner scan) {  
        super.nhap(); // Gọi phương thức nhập từ lớp cha  
        System.out.println(" Nhập nơi đến:");  
        this.noiDen = scan.nextLine();  
        System.out.println(" Nhập số ngày đi được:");  
        this.soNgayDiDuoc =  
        Float.parseFloat(scan.nextLine());  
    }  
}
```

output

- > Nhập mã số: A12
- > Nhập họ tên tài xế: CyberLearn
- > Nhập số xe: 51G-67.789
- > Nhập doanh thu: 80000
- > Nhập số tuyến: Số 8
- > Nhập số KM: 80.7

OVERRIDE & DẪN XUẤT

Subclass thừa hưởng được các method từ superclass

- Tuy nhiên **một số method** thừa hưởng được **không còn phù hợp nữa** như là không phù hợp hoàn toàn hoặc cách cài đặt hoàn toàn khác.
- Khi đó những method đó **cần cài đặt lại sao cho phù hợp với subclass**.

Để thực hiện **Override** nên thêm **@Override** trước method của subclass

@Override

```
public void input (){  
  
    ...  
}
```

Khi method Override cần dùng lại method superclass có thể gọi bằng cách **super.methodName (...)** của superclass

@Override


```
public void input(){  
  
    super.input();  
}
```

Nếu method của superclass có phạm vi là **private** thì **không thể Override**.

Nếu method của superclass có phạm vi là **protected** hoặc **public** thì có thể **Override**.

Nếu method của superclass có phạm vi là **protected** thì khi **Override** ở subclass có thể **chuyển thành** phạm vi **public**. Ngược lại thì không được

ĐA HÌNH (POLYMORPHISM)



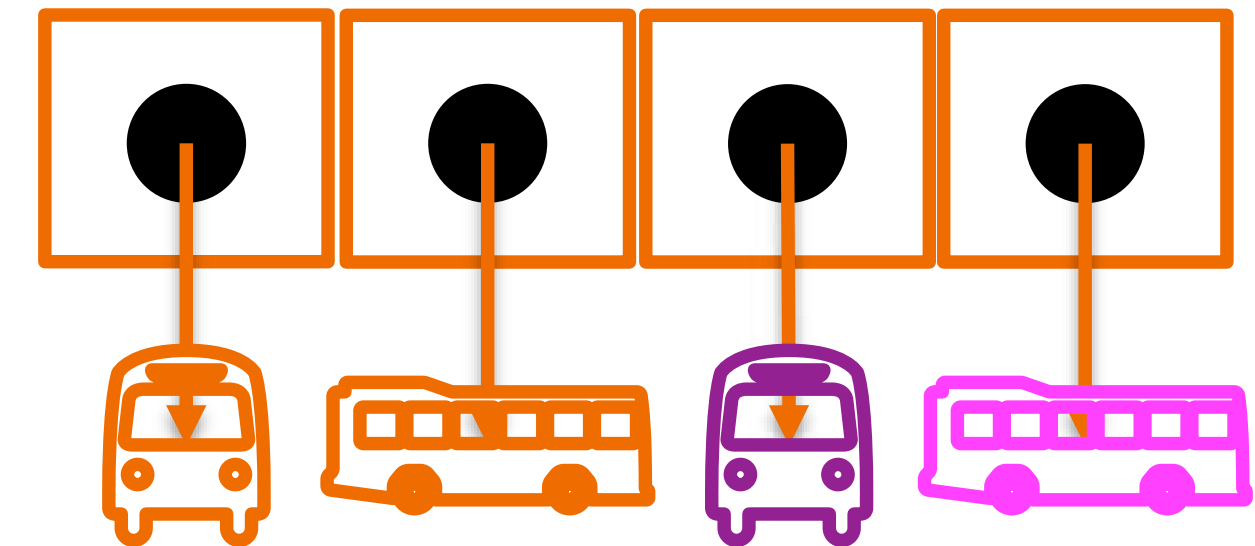
```
ArrayList<ChuyenXe> listChuyenXe = new  
ArrayList<ChuyenXe>();
```

```
ChuyenXe chuyenXe = new ChuyenXeNoiThanh();  
listChuyenXe.add(chuyenXe);
```

```
chuyenXe = new ChuyenXeNgoaiThanh();  
listChuyenXe.add(chuyenXe);
```

```
chuyenXe = new ChuyenXeLao();  
listChuyenXe.add(chuyenXe);
```

```
chuyenXe = new ChuyenXeCampuchia();  
listChuyenXe.add(cxCam);
```



Một object thuộc superclass có thể giữ reference
đến object thuộc các subclass bất kỳ cấp nào.

ĐA HÌNH (POLYMORPHISM)

```
ArrayList<ChuyenXe> listChuyenXe = new ArrayList<ChuyenXe>();
```

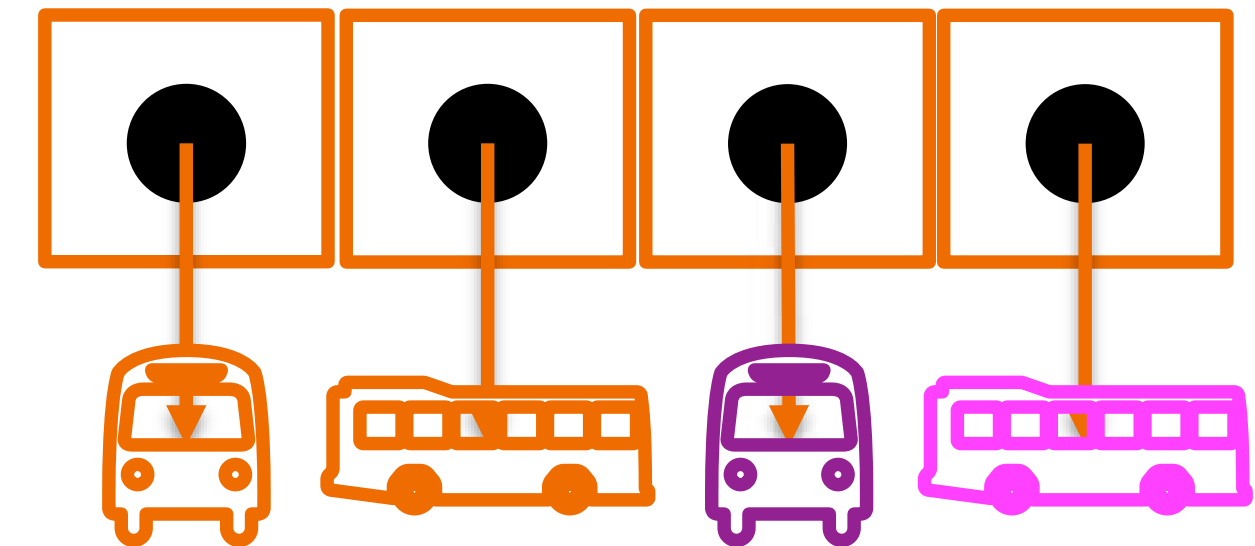
```
ChuyenXe chuyenXe = new ChuyenXeNoiThanh();  
listChuyenXe.add(chuyenXe);
```

```
chuyenXe = new ChuyenXeNgoaiThanh();  
listChuyenXe.add(chuyenXe);
```

```
chuyenXe = new ChuyenXeLao();  
listChuyenXe.add(chuyenXe);
```

```
chuyenXe = new ChuyenXeCampuchia();  
listChuyenXe.add(cxCam);
```

```
for (ChuyenXe cx: listChuyenXe){  
    this.tongDoanhThu += cx.tinhDoanhThu();  
}
```



Object thuộc superclass chỉ có thể **nhìn thấy trực tiếp** các method được cài đặt trong superclass nhưng **khi gọi** thì **thực sự gọi các method của subclass** nếu có **Override**

DEMO BÀI TOÁN XÀI TOÀN BỘ ĐỒNG GÓI + KẾ THỪA & ĐA HÌNH

Công ty du lịch V quản lý thông tin là các chuyến xe. Thông tin của 2 loại chuyến xe:

- ☐ * Chuyến xe nội thành: Mã số chuyến, Họ tên tài xế, số xe, số tuyến, số km đi được, doanh thu.
- ☐ * Chuyến xe ngoại thành: Mã số chuyến, Họ tên tài xế, số xe, nơi đến, số ngày đi được, doanh thu. Xây dựng chương trình quản lý các chuyến xe, tính tổng doanh thu của công ty và doanh thu từng loại xe



Tính doanh thu các cho công ty, cho ngoại thành và nội thành.

1. Đọc kỹ nghiệp vụ
2. Phân tích sơ đồ lớp, các cây kế thừa
3. Tổ chức lớp cha
4. Tổ chức lớp con
5. Xác định các phương thức Override
6. Xử lý các nghiệp vụ theo giải thuật

DỰ ÁN: QUẢN LÝ GIAO DỊCH

Xây dựng chương trình quản lý danh sách các giao dịch. Hệ thống quản lý 2 loại giao dịch:

* Giao dịch vàng: Mã giao dịch, ngày giao dịch (ngày, tháng, năm), đơn giá, số lượng, loại vàng. Thành tiền được tính như sau:

Thành tiền = số lượng * đơn giá.

*Giao dịch tiền tệ: Mã giao dịch, ngày giao dịch (ngày, tháng, năm), Đơn giá, số lượng, tỉ giá, loại tiền tệ có 3 loại: tiền Việt Nam, tiền USD, tiền Euro. Thành tiền được tính như sau:

- Nếu là tiền USD hoặc Euro thì: thành tiền = số lượng * đơn giá* tỉ giá

- Nếu là tiền VN thì: thành tiền = số lượng * đơn giá

Thực hiện các yêu cầu sau:

- ☐ Xây dựng các lớp với chức năng thừa kế.
- ☐ Nhập xuất danh sách các giao dịch.
- ☐ Tính tổng số lượng cho từng loại.
- ☐ Tính trung bình thành tiền của giao dịch tiền tệ.
- ☐ Xuất ra các giao dịch có đơn giá > 1 tỷ.

DỰ ÁN : QUẢN LÝ HỌC VIỆN NGHIÊN CỨU & GIẢNG DẠY

Giả sử cần xây dựng chương trình quản lý dùng cho một học viện nghiên cứu giảng dạy và ứng dụng. Đối tượng quản lý bao gồm các sinh viên đang theo học, các nhân viên đang làm việc tại học viện, các khách hàng đến giao dịch mua bán sản phẩm ứng dụng. Dựa vào một số đặc tính của từng đối tượng, người quản lý cần đưa ra cách thức đánh giá khác nhau.

Vậy hãy xây dựng các lớp sau:

- Lớp *Person*: bao gồm các thuộc tính họ tên, địa chỉ, mã, email
- Các lớp *Student*, *Employee*, *Customer* (mô tả dưới đây) thừa kế lớp *Person*.
 - Lớp *Student*: bao gồm các thuộc tính toán, lý, hóa.
 - Lớp *Employee*: bao gồm thuộc tính: số ngày làm việc, lương theo ngày.
 - Lớp *Customer*: bao gồm thuộc tính tên công ty, trị giá hóa đơn, đánh giá
 - Lớp *ListPerson* để quản lý các đối tượng trên

Chương trình cho phép thực hiện:

- 1) Thêm một người vào danh sách
- 2) Xóa 1 người khỏi danh sách theo mã
- 3) Sắp xếp danh sách theo thứ tự họ tên

DỰ ÁN : QUẢN LÝ HỌC VIỆN NGHIÊN CỨU & GIẢNG DẠY

Giải thuật

- Bước 1 : $i = 0$; // bắt đầu từ đầu dãy
- Bước 2 : $j = i + 1$; // tìm các phần tử $a[j] < a[i], j > i$
- Bước 3 :
Trong khi $j \leq N$ thực hiện
Nếu $a[j] < a[i]$: Hoán vị $a[i], a[j]$;
 $j = j + 1$;
- Bước 4 : $i = i + 1$;
Nếu $i < N$: Lặp lại Bước 2.
Ngược lại: Dừng.

```
void interchangeSort(int []a, int N )
{
    int i, j;
    for (i = 0 ; i < N - 1 ; i++)
    {
        for (j = i + 1; j < N ; j++)
            if(a[j] < a[i])
            {
                int tam = a[i];
                a[i] = a[j];
                a[j] = tam;
            }
    }
}
```



Sorting Algorithms



BÀI TẬP NỘP : QUẢN LÝ NHÀ ĐẤT

Xây dựng chương trình quản lý danh sách các giao dịch nhà đất. Thông tin bao gồm:

- ☐ * Giao dịch đất: Mã giao dịch, ngày giao dịch (ngày, tháng, năm), đơn giá, loại đất (loại A, B, C), diện tích.
 - Nếu là loại B, C thì: thành tiền = diện tích * đơn giá.
 - Nếu là loại A thì: thành tiền = diện tích * đơn giá * 1.5
- ☐ *Giao dịch nhà: Mã giao dịch, ngày giao dịch (ngày, tháng, năm), đơn giá, loại nhà (cao cấp, thường), địa chỉ, diện tích.
 - Nếu là loại nhà cao cấp thì: thành tiền = diện tích * đơn giá.
 - Nếu là loại thường thì: thành tiền = diện tích * đơn giá * 90%

Thực hiện các yêu cầu sau:

- ☐ - Xây dựng các lớp với chức năng thừa kế.
 - Nhập xuất danh sách các giao dịch.
 - Tính tổng số lượng cho từng loại.
 - Tính trung bình thành tiền của giao dịch đất. ☐ Xuất ra các giao dịch của tháng 9 năm 2013.

BÀI TẬP NỘP : QUẢN LÝ DANH SÁCH HÓA ĐƠN TIỀN ĐIỆN

Xây dựng chương trình quản lý danh sách hoá đơn tiền điện của khách hàng. Thông tin bao gồm các loại khách hàng :

- ☐ * Khách hàng Việt Nam: mã khách hàng, họ tên, ngày ra hoá đơn (ngày, tháng, năm), đối tượng khách hàng (sinh hoạt, kinh doanh, sản xuất): số lượng (số KW tiêu thụ), đơn giá, định mức. Thành tiền được tính như sau:
 - Nếu số lượng \leq định mức thì: thành tiền = số lượng * đơn giá.
 - Ngược lại thì: thành tiền = số lượng * đơn giá * định mức + số lượng KW vượt định mức * Đơn giá * 2.5.
- ☐ * Khách hàng nước ngoài: mã khách hàng, họ tên, ngày ra hoá đơn (ngày, tháng, năm), quốc tịch, số lượng, đơn giá. Thành tiền được tính = số lượng * đơn giá.

Thực hiện các yêu cầu sau:

- ☐ - Xây dựng các lớp với chức năng thừa kế.
 - Nhập xuất danh sách các hóa đơn khách hàng.
 - Tính tổng số lượng cho từng loại khách hàng.
 - Tính trung bình thành tiền của khách hàng người nước ngoài.
 - Xuất ra các hoá đơn trong tháng 09 năm 2013 (của cả 2 loại khách hàng).

BÀI TẬP NỘP : QUẢN LÝ KHÁCH SẠN

Một khách sạn X cần quản lý các hóa đơn của khách hàng thuê phòng. Hóa đơn có 2 loại: hóa đơn theo giờ, hóa đơn theo ngày . Thông tin chung của chi tiết hóa đơn là: Mã hóa đơn, ngày hóa đơn (ngày, tháng, năm), Tên khách hàng, mã phòng, đơn giá. Thông tin riêng của từng loại hóa đơn gồm:

- ☐ * Hóa đơn theo giờ còn có số giờ thuê. Thành tiền = số giờ thuê * đơn giá. Nếu trường hợp số giờ > 24 tiếng và < 30 tiếng thì cũng chỉ tính 24 giờ. Nếu trường hợp số giờ là > 30 tiếng thì không dùng loại hóa đơn theo giờ.
- ☐ * Hóa đơn theo ngày sẽ có số ngày thuê. Thành tiền = số ngày thuê * đơn giá. Nếu số ngày >7 thì giảm 20% đơn giá cho những ngày còn lại.

Thực hiện các yêu cầu sau:

- ☐ 1) Xây dựng các lớp với chức năng thừa kế.
- ☐ 2) Nhập xuất danh sách các hóa đơn thuê phòng.
- ☐ 3) Tính tổng số lượng cho từng loại thuê phòng.
- 4) ☐ Tính trung bình thành tiền của hóa đơn thuê phòng trong tháng 9/2013.

BÀI CUỐI KHÓA : QUẢN LÝ NHÂN SỰ

Xây dựng ứng dụng Quản lý nhân sự của 1 công ty bằng với các yêu cầu sau:

Công ty có tên công ty, mã số thuế, doanh thu tháng. Công ty có 3 loại nhân viên: giám đốc, trưởng phòng, nhân viên thường. Mỗi người trong công ty phải có các thông tin: mã số, họ tên, số điện thoại, số ngày làm việc, lương 1 ngày và cách tính lương. Ngoài các thông tin chung, mỗi chức vụ trong công ty còn có các thuộc tính riêng:

Nhân viên:

- Có thêm trưởng phòng quản lý. Nếu không có ai quản lý thì để null
- Công thức tính lương tháng : lương 1 ngày * số ngày làm việc .Lương 1 ngày của nhân viên: 100

Trưởng phòng:

- Có số lượng nhân viên dưới quyền. Thuộc tính này tăng lên khi có thêm 1 nhân viên thêm vào do trưởng phòng đó quản lý.
- Công thức tính lương tháng: lương 1 ngày * số ngày làm việc + 100 * số lượng nhân viên dưới quyền. Lương 1 ngày của trưởng phòng: 200

Giám đốc:

- Có thêm thuộc tính cổ phần trong công ty. Trị số này là số %, không được vượt quá 100%
- Công thức tính lương tháng : lương 1 ngày * số ngày làm việc. Lương 1 ngày của Giám đốc: 300

In ra menu cho chọn các chức năng sau: (Xuất thông tin theo biểu mẫu tablet có cột số thứ tự)

1. Nhập thông tin công ty
2. Phân bổ Nhân viên vào Trưởng phòng
3. Thêm, xóa thông tin một nhân sự (có thể là Nhân viên, trưởng phòng hoặc giám đốc). Lưu ý khi xóa trưởng phòng, phải ngắt liên kết với các nhân viên đang tham chiếu tới.
4. Xuất ra thông tin toàn bộ người trong công ty
5. Tính và xuất tổng lương cho toàn công ty
6. Tìm Nhân viên thường có lương cao nhất
7. Tìm Trưởng Phòng có số lượng nhân viên dưới quyền nhiều nhất
8. Sắp xếp nhân viên toàn công ty theo thứ tự abc
9. Sắp xếp nhân viên toàn công ty theo thứ tự lương giảm dần
10. Tìm Giám Đốc có số lượng cổ phần nhiều nhất
11. Tính và Xuất tổng THU NHẬP của từng Giám Đốc

Thu nhập = Lương tháng + số cổ phần * Lợi nhuận công ty

@Lợi nhuận công ty = Doanh thu tháng của công ty - tổng lương toàn công ty trong tháng.

XÓA NHÂN SỰ

Mã số: 1	Họ tên: Lan	SĐT: 09832	Số ngày làm: 20.0	Lương: 2000.0	Mã TP: 8	Tên TP:TP Mai
Mã số: 2	Họ tên: Hưng	SĐT: 098232	Số ngày làm: 23.0	Lương: 2300.0	Mã TP: 8	Tên TP:TP Mai
Mã số: 3	Họ tên: Việt	SĐT: 098132	Số ngày làm: 31.0	Lương: 3100.0	Mã TP: 9	Tên TP:TP Luân
Mã số: 4	Họ tên: An	SĐT: 091832	Số ngày làm: 25.0	Lương: 2500.0	Mã TP: 9	Tên TP:TP Luân
Mã số: 5	Họ tên: Tuyết	SĐT: 098432	Số ngày làm: 23.0	Lương: 2300.0	Mã TP: 10	Tên TP:TP Kiệt
Mã số: 6	Họ tên: Mỹ	SĐT: 092832	Số ngày làm: 31.0	Lương: 3100.0	Chưa phân bổ	
Mã số: 8	Họ tên: TP Mai	SĐT: 098233	Số ngày làm: 24.0	Số Nhân viên:2	Lương: 4800.0	
Mã số: 9	Họ tên: TP Luân	SĐT: 091833	Số ngày làm: 21.0	Số Nhân viên:2	Lương: 4200.0	
Mã số: 10	Họ tên: TP Kiệt	SĐT: 091833	Số ngày làm: 23.0	Số Nhân viên:1	Lương: 4600.0	
Mã số: 10	Họ tên: GD Tiên	SĐT: 0981	Số ngày làm: 19.0	Lương: 5700.0		
Mã số: 11	Họ tên: GD Huệ	SĐT: 09181	Số ngày làm: 21.0	Lương: 6300.0		
Mã số: 14	Họ tên: nv thường	SĐT: 091	Số ngày làm: 25.0	Lương: 2500.0	Chưa phân bổ	
Mã số: 15	Họ tên: tphuong mới	SĐT: 123	Số ngày làm: 26.0	Số Nhân viên:0	Lương: 5200.0	

GIẢI THUẬT XÓA NHÂN SỰ

Xóa nhân sự

Kiểm tra tồn tại

- Kiểm tra mã nhân viên cần xóa có trong danh sách nhân sự hay không
- Nếu có, thực hiện các Trường hợp bên dưới

Nếu Trưởng phòng

1. Duyệt danh sách nhân sự và kiểm tra nếu là nhân viên thường
2. Kiểm tra Trưởng phòng trong NV Thường khác null
3. Kiểm tra mã TP của NV đang duyệt và mã TP cần xóa, nếu bằng nhau
4. Gán Trưởng phòng trong NVThuong là null
5. Xóa Trưởng phòng
 - Xóa khỏi list Nhân Sự
 - Xóa khỏi list Trưởng phòng

Nếu là Nhân viên thường

1. Lấy Trưởng phòng đang quản lý
2. Kiểm tra khác null
3. Tìm Trưởng phòng trong DS Trưởng phòng có mã TP bằng với TP trong bước 1
4. Giảm số lượng nhân viên của Trưởng phòng đang quản lý
5. Xóa Nhân viên thường khỏi danh sách

Nếu là giám đốc

- Xóa bình thường