

# TƯ DUY LẬP TRÌNH

CYBERSOFT.EDU.VN

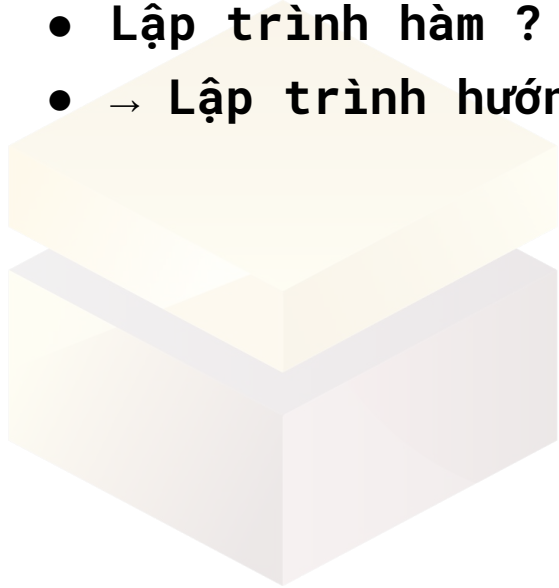
[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Kiến thức về lớp - đối tượng

1. Tại sao cần lớp, đối tượng
2. Bài toán thực tế → phần mềm
3. Các từ khóa trong lớp, đối tượng
4. Tạo lớp
5. Các hàm khởi tạo mặc định
6. Chồng hàm
7. Thể hiện của lớp (Đối tượng)
8. Phạm vi truy xuất, dẫn xuất, các từ khóa (private, protected, public) → Lấy ví dụ công thức Phở của dòng họ, tiền bạc của mỗi gia đình.
9. Has-a, Is-a

# Tại sao cần lớp, đối tượng

- Lập trình dòng lệnh ?
- Lập trình hàm ?
- → Lập trình hướng đối tượng ?



CYBERSOFT

ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Ví dụ thực tế

Tên hàng

Cách bảo  
quản

Giá cả

Khối lượng

Thành phần



# Ví dụ thực tế

- Một mặt hàng có những thứ đi kèm với nó: tên, giá cả, công thức, hạn sử dụng ...
- Mà không phải chỉ có 1 mặt hàng mà có rất nhiều loại khác nhau.
- Để dễ dàng cho việc quản lý cũng như tổ chức. → làm nên một đối tượng: HangHoa

CYBERSOFT  
ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH  
[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Đối tượng

- Đối tượng là một thực thể phản ánh đúng vật đó trong đời sống, và được mô tả trong lập trình.
- Một đối tượng có 2 thứ cần lưu ý: **Thuộc Tính + Phương thức**
- Thuộc tính là những gì thuộc về đối tượng đó, mô tả cho đối tượng đó.
- Phương thức là những hành động mà đối tượng có thể làm được, luôn nhớ là chỉ liên quan đến đối tượng này mà thôi.
- 

CYBERSOFT  
ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Ví dụ về đối tượng

Đối tượng	Thuộc tính / Dữ liệu	Hành vi/ Phương thức
Phân số	<ul style="list-style-type: none"><li>- Tử</li><li>- Mẫu</li></ul>	<ul style="list-style-type: none"><li>- Cộng</li><li>- Trừ</li><li>- Nhân</li><li>- Chia</li><li>- ....</li></ul>
Xe	<ul style="list-style-type: none"><li>- Màu</li><li>- Kích thước</li><li>- Trọng lượng</li><li>- Mẫu</li><li>- Nhà sản xuất</li><li>- .....</li></ul>	<ul style="list-style-type: none"><li>- Khởi động máy</li><li>- Tăng tốc</li><li>- Tắt máy</li><li>- ....</li></ul>

# Ví dụ về đối tượng (tt)

Đối tượng	Thuộc tính / Dữ liệu	Hành vi/ Phương thức
<b>Tài khoản</b>	<ul style="list-style-type: none"><li>- Tên tài khoản</li><li>- Người sở hữu</li><li>- Số tài khoản</li><li>- Loại tài khoản</li><li>- Số dư</li><li>- .....</li></ul>	<ul style="list-style-type: none"><li>- Hiển thị số dư hiện tại</li><li>- Rút tiền mặt</li><li>- Thống kê</li><li>- Đóng tài khoản</li><li>- ....</li></ul>
<b>Nhân viên</b>	<ul style="list-style-type: none"><li>- Mã nhân viên</li><li>- Họ</li><li>- Tên</li><li>- CMND</li><li>- Giới tính</li><li>- .....</li></ul>	<ul style="list-style-type: none"><li>- Hiển thị thông tin nhân viên</li><li>- Thay đổi thông tin nhân viên</li><li>- ....</li></ul>



# Ví dụ về đối tượng (tt)

Đối tượng	Thuộc tính / Dữ liệu	Hành vi/ Phương thức
<b>Sinh viên</b>	<ul style="list-style-type: none"><li>- Mã sinh viên</li><li>- Tên</li><li>- Họ</li><li>- Giới tính</li><li>- Địa chỉ</li><li>- .....</li></ul>	<ul style="list-style-type: none"><li>- Thay đổi thông tin</li><li>- Hiện thị thông tin</li><li>- Lấy điểm trung bình học kì</li><li>- ....</li></ul>
<b>Trường học</b>	<ul style="list-style-type: none"><li>- Mã trường</li><li>- Tên trường</li><li>- Địa chỉ</li><li>- .....</li></ul>	<ul style="list-style-type: none"><li>- Hiện thị thông tin trường</li><li>- Đếm sinh viên trường</li><li>- ....</li></ul>

# Lớp đối tượng

- Chúng ta không thể chỉ có một đối tượng. Ví dụ: Xe (ĐT) thì có nhiều xe, nhà (1 ĐT) cũng có nhiều nhà...  
→ 1 tập hợp các đối tượng, khuôn mẫu/template → **LỚP ĐỐI TƯỢNG**
- Một số từ khóa cần nhớ :
  - Đối tượng : **Object**
  - Lớp đối tượng : **Class**
  - Thuộc tính/Dữ liệu/Biến thành viên : **Attributes/ Data members**
  - Phương thức (**Method**)

[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Tạo lớp đối tượng

```
public class MyPoint
{
    public double x;
    public double y;
}
```


- Tạo lớp sử dụng từ khóa **Class**
- Tên Lớp **MyPoint**
- Có hai thuộc tính/biến thành viên x, y
- **public** : Dẫn xuất / modifier (Trình bày sau)

TẠO CHUYÊN GIA LẬP TRÌNH  
[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Demo chạy

```
public class TestMyPoint2      1
{                                2
    public static void main(String[] args)  3
    {                                4
        MyPoint p = new MyPoint();      5
        MyPoint q = new MyPoint();      6
        p.x = 2;                        7
        p.y = 3;                        8
        q.x = 0.5;                      9
        q.y = -0.5;                    10
        System.out.println("(" + p.x + "," + p.y + ")"); 11
        System.out.println("(" + q.x + "," + q.y + ")"); 12
    }                                13
}                                    14
```


# Instance - Thể hiện đối tượng



**Class: Teacher**


Name[firstName, lastName]  
Age  
Gender  
Interests  
Bio{ "[Name] is [Age] years old. They like [Interests]." }  
Subject  
Greeting{ "Hello. My name is [Prefix][lastName], and I teach [Subject]." }

Instantiation



**Object: teacher1**

Name[Dave, Griffiths]  
Age: 31  
Gender: Male  
Interests: football, cookery  
Bio{ "Dave Griffiths is 31 years old. They like football and cookery." }  
Subject: Math  
Greeting{ "Hello. My name is Mr. Griffiths, and I teach math." }



**Object: teacher2**

Name[Melanie, Hall]  
Age: 26  
Gender: Female  
Interests: playing guitar, archery  
Bio{ "Melanie Hall is 26 years old. They like playing guitar and archery." }  
Subject: Physics  
Greeting{ "Hello. My name is Ms. Hall, and I teach physics." }

Class Name

Keyword

```
Student student1 = new Student();
```

Object Name

Constructor

# Thể hiện của lớp đối tượng / Instance

1. Muốn xài được lớp đối tượng → Phải tạo các **đối tượng cụ thể**. Một trường hợp cụ thể của lớp đối tượng đối tượng → Thể hiện của lớp đối tượng (**Instance** of Class)
2. Sử dụng từ khóa **new** để tạo ra thể hiện của lớp đối tượng

CYBERSOFT

ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Quay lại ví dụ MyPoint

Trong dòng 5, 6, chúng ta tạo ra 2 đối tượng *p*, *q* sử dụng từ khóa **new**

- Mỗi đối tượng là một thể hiện của lớp **MyPoint**


```
MyPoint p = new MyPoint();  
MyPoint q = new MyPoint();
```

- Dòng 7, 8, 9, 10 chúng ta truy xuất đến 2 thuộc tính của lớp và gán giá trị cho nó. **SỬ DỤNG TOÁN TỬ CHẤM (.)**

```
p.x = 2;  
p.y = 3;
```

```
q.x = 0.5;  
q.y = -0.5;
```

# Thay đổi private cho lớp MyPoint



```
public class MyPoint
{
    private double x;
    private double y;
}
```



# Dẫn xuất ( Access Modifier )

- **private** : chỉ được xài trong lớp đó, ra ngoài lớp không được xài
- **public** : Xài nơi nào cũng được
- **protected**: Các lớp con cháu dòng họ được xài (Kế thừa sẽ thấy)
- Lưu ý: Các biến thành viên của lớp đối tượng thường để **private** → che dấu thông tin của lớp đó.

CYBERSOFT  
ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Tính đóng gói (Encapsulation )

PhanSo
-tuSo : int -mauSo : int
+getTuSo() : int +setTuSo(in tuSo : int) : void +getMauSo() : int +setMauSo(in mauSo : int) : void

# Tính đóng gói (Encapsulation )

- Các **field** lưu trữ **data** của class.
- Các **method** định nghĩa các **chức năng** của class
  - Thường **tương tác** với các **field**.
- **Encapsulation** là một trong các tính chất nền tảng của OOP (**Data hiding**)
  - **Developer** có thể **hide/encapsulate** một số các **field**, các **method** của class.
  - Đồng thời có thể **expose** các **field**, **method** còn lại.

# Tính đóng gói (Encapsulation )

- Khi **sử dụng** một **class**, có thể xem class đó như là một **black box**
  - Chỉ **quan tâm** đến các **field** và các **method** cần dùng.
  - **Không quan tâm** sự **phức tạp bên trong**.
- Ví dụ khi sử dụng method getTuSo của class PhanSo
  - Không quan tâm method được cài đặt như thế nào.
  - Chỉ quan tâm cách sử dụng.

# Tính đóng gói (Encapsulation )

- Encapsulation giúp **dễ dàng thay đổi code** bên trong phương thức của class mà **không ảnh hưởng đến các class khác** đang sử dụng nó.
- Điều này giúp **dễ dàng nâng cấp** ứng dụng bằng cách **thay đổi đúng các class cần nâng cấp**.



# Phương thức

- Một số loại method thường dùng
  - (1) Cung cấp cơ chế **get / set** các field trong class. Còn được gọi là các **Accessor**.
    - getTuSo, setTuSo, ...
  - (2) **Nhập xuất** thông tin ra màn hình **console**
    - xuất(), nhập(),...
  - (3) Xử lý các **nghiệp vụ**
    - cong(PhanSo ps), tru(PhanSo ps),...

# Phương thức

- **Cho phép** các **class khác truy xuất** trực tiếp phương thức, sử dụng **public keyword**.
- **Ngăn** các **class khác truy xuất** trực tiếp, sử dụng **private keyword**.
- Các **method** thường **tương tác** với các **field** bên trong **class**.

# Phương thức / Truy xuất thuộc tính

```
public class MyPoint      1
{                          2
    // thuộc tính/data members  3
    private double x;      4
    private double y;      5
                            6
    // accessor methods (phương thức truy xuất) 7
    public double getX(){   8
        return x;          9
    }                      10
    public double getY(){  11
        return y;          12
    }                      13
                            14
```



# Phương thức / Thiết lập giá trị cho thuộc tính

```
// mutator methods (Phương thức thiết lập) 15
public void setX(double x){                16
    this.x = x;                             17
}                                           18
public void setY(double y){                19
    this.y = y;                             20
}                                           21
                                           22
```

**this** → con trỏ của **thể hiện (instance)** của lớp hiện hành

# Một số thuộc tính khác

```
// một số method khác 23
public void setLocation(double x, double y){ 24
    this.x = x; 25
    this.y = y; 26
} 27
public double distanceTo(MyPoint p){ 28
    double diffXSquare = Math.pow((p.getX()-x),2); 29
    double diffYSquare = Math.pow ((p.getY()-y),2); 30
    return Math.sqrt(diffXSquare+diffYSquare); 31
} 32
public String toString(){ 33
    return "("+x+", "+y+")"; 34
} 35
} 36
```

# Sử dụng toàn bộ lớp MyPoint

```
public class TestMyPoint3      1
{                                2
    public static void main(String[] args)  3
    {                                4
        MyPoint p = new MyPoint();      5
        MyPoint q = new MyPoint();      6
        p.setX(6.0);                    7
        p.setY(5.0);                    8
        q.setLocation(p.getX(),p.getY());
        9
        System.out.println("q="+q);      10
        p.setLocation(10.0,2.0);          11
        System.out.print("Distance from "+p+" to ");  12
        System.out.println(q+" is "+p.distanceTo(q));  13
    }                                    14
}                                       15
```



FT  
RINH

# Quá tải hàm/ Chồng hàm / Overloading

- Các hàm cùng tên nhưng khác tham số
- Hữu ích để tạo nhiều version khác nhau của hàm

// chồng hàm

```
public MyPoint(){  
    this(1.0,1.0);  
}
```

```
public MyPoint(double x,double y){  
    this.x = x;  
    this.y = y;  
}
```

```
public MyPoint(MyPoint p){  
    this(p.getX(),p.getY());  
}
```

# Phương thức / Hàm khởi tạo (Constructor)

- **Constructor** là một Method đặc biệt được sử dụng để khởi tạo đối tượng (Object)
- **Constructor** được gọi tại thời điểm object được tạo
- Phải cùng tên với tên class
- Không có kiểu trả về
- Có thể có tham số hoặc không có tham số
- 

CYBERSOFT

ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Phương thức / Hàm khởi tạo (Constructor)

```
public MyPoint(){  
    this(1.0,1.0);  
}
```

```
public MyPoint(double x,double y){  
    this.x = x;  
    this.y = y;  
}
```

```
public MyPoint(MyPoint p){  
    this(p.getX(),p.getY());  
}
```

# Dự án 11: Tính chu vi và diện tích hình chữ nhật

Xây dựng lớp đối tượng Hình chữ nhật và cài đặt các phương thức tính chu vi và diện tích của HCN.



# CYBERSOFT

ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)



# Lớp đối tượng + hàm khởi tạo

//Hàm khởi tạo không tham số

```
public HCN() {  
    chieuDai = 5;  
    chieuRong = 4;  
}
```

//Hàm khởi tạo không tham số

```
public HCN() {  
}
```

//Hàm khởi tạo có tham số

```
public HCN(float _chieuD, float _chieuR)  
{  
    chieuDai = _chieuD;  
    chieuRong = _chieuR;  
}
```

Hàm khởi tạo có những đặc điểm sau:

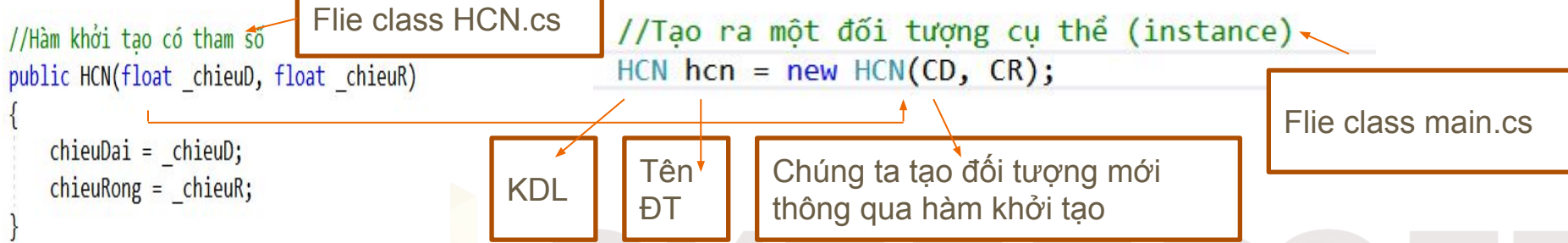
- + Có tên hàm trùng với tên của lớp đối tượng
- + Được public ra ngoài để những lớp khác có thể sử dụng được

\* Hàm khởi tạo không tham số: Có thể **để trống** hoặc gán **giá trị mặc định** cho thuộc tính của lớp đối tượng đó.

\* Hàm khởi tạo có tham số: Khi tạo ra lớp đối tượng mới, mình truyền trực tiếp giá trị cho thuộc tính



# Lớp đối tượng + hàm chồng



Hàm chồng → 2 hàm trùng tên nhưng phải khác tham số.  
Khi sử dụng thì hàm nào được gọi ra thì sẽ đi vào đó và thực hiện câu lệnh. Ở trên ví dụ hàm chồng cho 2 hàm khởi tạo. Một hàm không có tham số, và một hàm khởi tạo có 2 tham số.

# Lớp đối tượng + Phương thức

Flie class HCN.cs

```
//Method: Phương thức  
public float tinhChuVi()  
{  
    return ((chieuDai + chieuRong) * 2);  
}
```

Phương thức đại biểu cho hành động mà đối tượng đó có thể thực hiện được. Ở đây viết phương thức chính chu vi cho class HCN;.

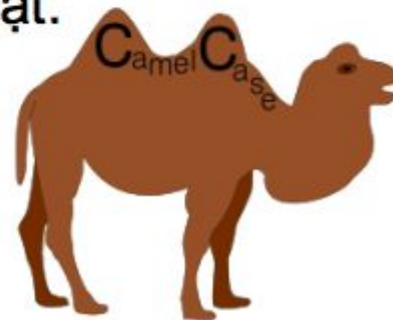
```
//Tạo ra một đối tượng cụ thể (instance)  
HCN hcn = new HCN(CD, CR);  
  
hcn.tinhChuVi();
```

Flie class main.cs

Khi muốn gọi phương thức ra chỉ cần “.tenphuongthuc()”. Tại sao phương tính chu vi mà file main lại thực hiện được. Do đã để trạng thái **public** ở trên phương thức

# Java Naming Convention

- Sử dụng chuẩn **Java naming convention** giúp **code dễ đọc** cho
  - Chính Developer đang làm.
  - Các Developer khác.
- **Tính dễ đọc** là một tính chất **rất quan trọng** trong chương trình Java
  - Dành ít thời gian để tìm hiểu code đã cài đặt.
- **Java Naming** tuân thủ theo **CamelCase**.



# Java Naming Convention

- **Class** bắt đầu bằng chữ hoa và là một danh từ
  - Phan**S**o, Hoc**S**inh, Sinh**V**ien,....
- **Field** bắt đầu bằng chữ thường
  - tu**S**o, mau**S**o, ho**V**a**T**en,...
- **Method** bắt đầu bằng chữ thường
  - tinh**L**uong(), cong(), chuyen**L**op(),...

# Java Naming Convention

- **Package** đặt bằng chữ thường

- org.nhanh, java.lang, ...

- Sử dụng { } theo luật sau

```
... {  
    ...  
    ...  
}
```

# Bài 1: Quản lý sinh viên - HĐT

Xây dựng chương trình THEO HƯỚNG ĐỐI TƯỢNG cho phép người dùng nhập vào: Tên, Mã SV, điểm Toán, Lý Hóa. Cho phép nhập nhiều Sinh viên và thực hiện:

- Tính điểm trung bình từng sinh viên  $(T + L + H)/3$
- Xếp loại từng sinh viên theo:  $\geq 9 \rightarrow$  Xuất Sắc,  $9 > \text{Giỏi} \leq 8$ ,  $8 > \text{Khá} \leq 7$ ,  $7 > \text{TB Khá} \leq 6$ ,  $6 > \text{TB} \leq 5$ , còn lại Yếu.
- In ra SV có ĐTB cao nhất.
- In ra tất cả sinh viên Yếu.
- Tìm sinh viên theo tên
- Tìm sinh viên theo mã
- Xóa 1 sinh viên

# LỚP - THUỘC TÍNH - PHƯƠNG THỨC

- Từ khóa : **class**
- Thuộc tính: Thuộc về lớp đối tượng
- Phương thức: Hành vi của lớp đối tượng



CYBERSOFT  
ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

THUỘC TÍNH

[HTTPS://CYBERSOFT.VN](https://cybersoft.vn)

PHƯƠNG THỨC

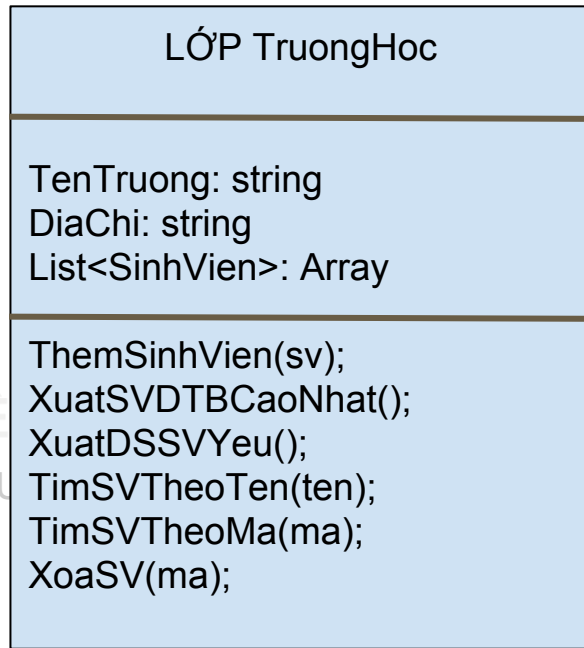
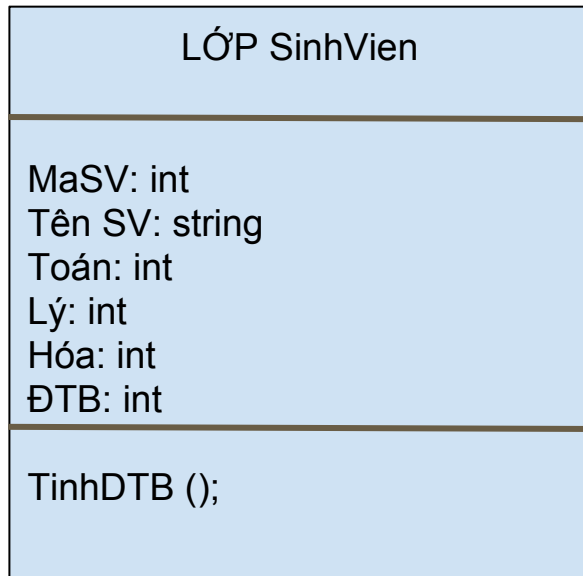
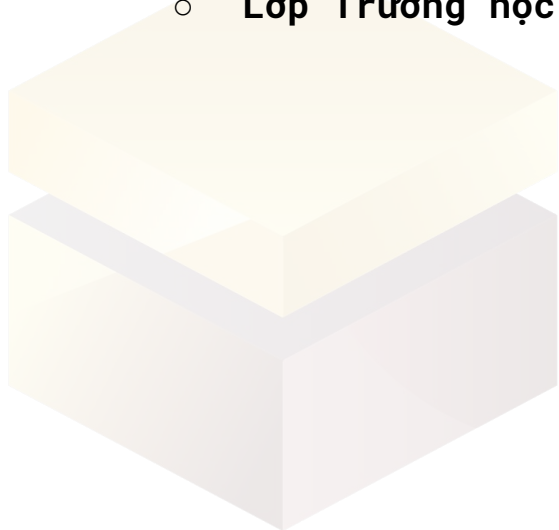
LỚP SINH VIÊN

MaSV: int  
Tên SV: string  
Toán: int  
Lý: int  
Hóa: int  
ĐTB: int

TinhDTB ();  
XepLoai();

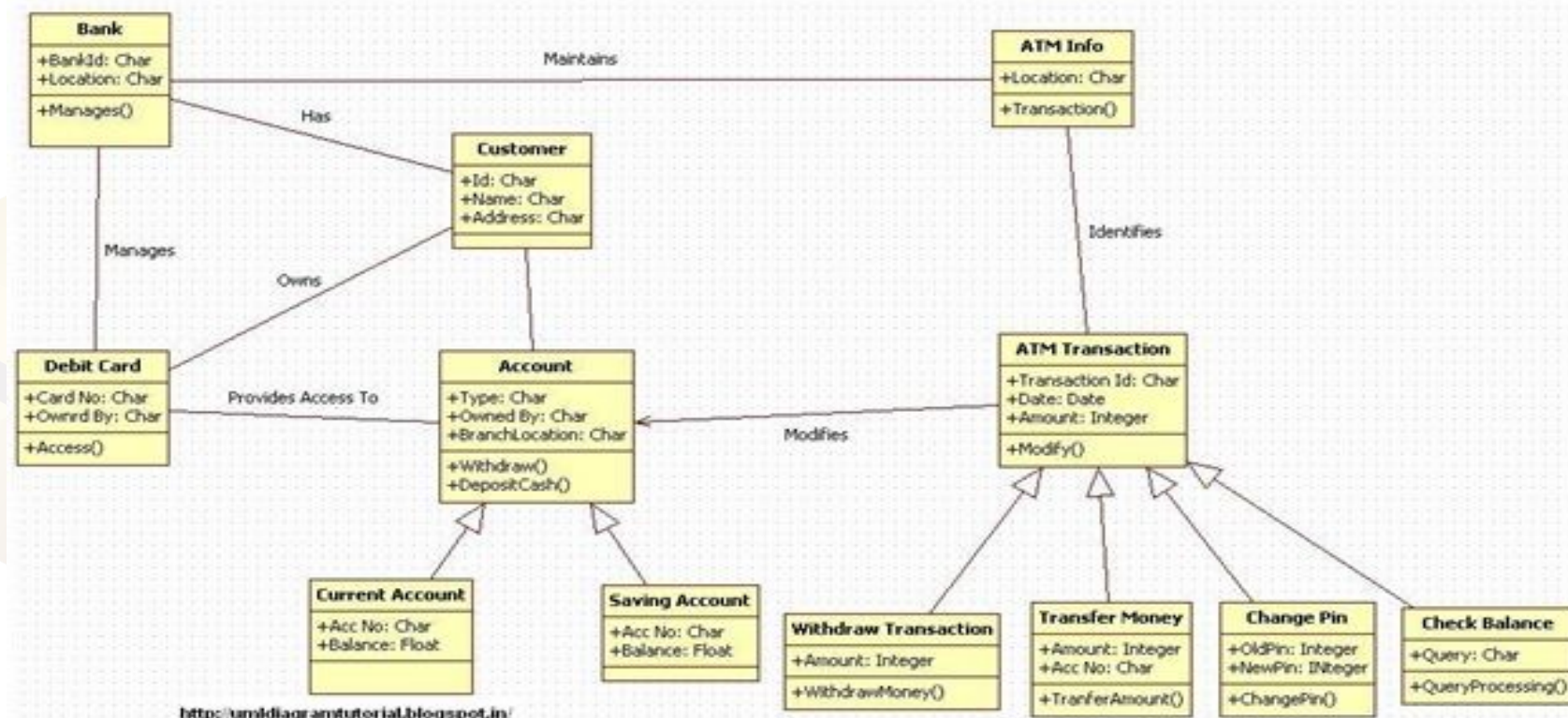
# Bước 1: Phân tích các lớp có trong dự án

- Có 2 lớp:
  - Lớp Sinh viên
  - Lớp Trường học





# SƠ ĐỒ LỚP



# Bước 2: Thiết kế giao diện



CYBERSOFT

ĐÀO TẠO CHUYÊN GIA LẬP TRÌNH

[HTTPS://CYBERSOFT.EDU.VN](https://cybersoft.edu.vn)

# Bài 2 : Ứng dụng nhân viên version 1

Công ty ABC cần xây dựng ứng dụng quản lý thông tin và tính lương cho nhân viên. Thông tin mỗi nhân viên bao gồm: mã nhân viên, họ tên, ngày sinh, địa chỉ.

- Lương nhân viên: hệ số lương \* lương cơ bản

## Yêu cầu:

1. Khai báo các lớp cần thiết.
2. Cài đặt 5 constructor cho lớp nhân viên
3. Cài đặt tất cả getter và setter cho tất cả các thuộc tính. Lưu ý, thực hiện các bước kiểm tra dữ liệu, trước khi gán giá trị thuộc tính trong các setter.
4. Hãy cài đặt lớp CongTy để quản lý danh sách các nhân viên trong công ty, với các chức năng sau:
  - a. Nhập, xuất danh sách các nhân viên.
  - b. Tính tổng tiền lương của tất cả nhân viên.
  - c. Xuất nhân viên có lương cao nhất.

