

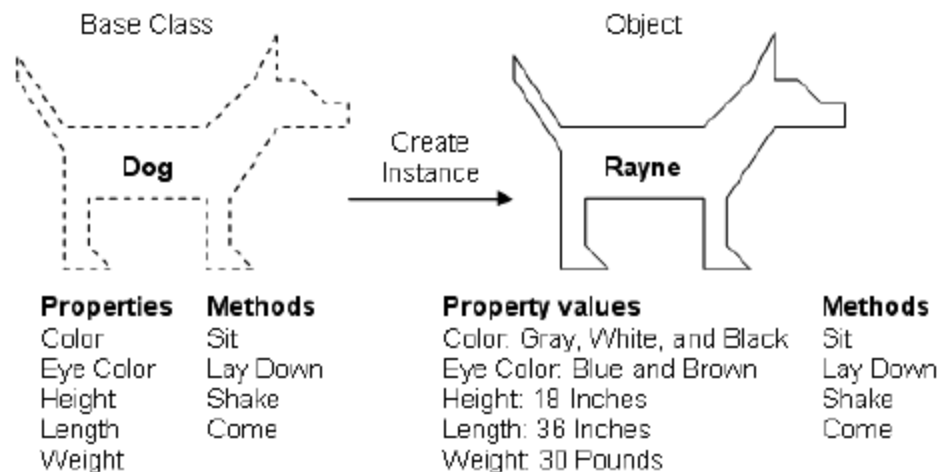


Bài 3. Trừu tượng hoá, đóng gói và xây dựng lớp

Lý thuyết và ngôn ngữ hướng đối tượng
(bài tập)

Class and Object

- A group of objects with similar properties and behaviors is described by a class.
- Class is a blueprint for an object.
- A class contains fields and methods that define the behavior of the object.
- A class is a description of a kind of object.



Class and Objects

Account Class (model of objects)

Definition of member variable

Attribute

String name;
long balance;

Method definition

operation

void deposit(int money);

void withdraw();

int checkBalance();

Account object of Mr Duc Binh

Attribute

name: Duc Binh
Balance: 2.000.000 VND

operation

Deposit()

Withdraw()

checkBalance()

INSTANTIATE

Account object of Mrs Thu Lan

Attribute

name: Thu Lan
Balance: 1.000.000 VND

operation

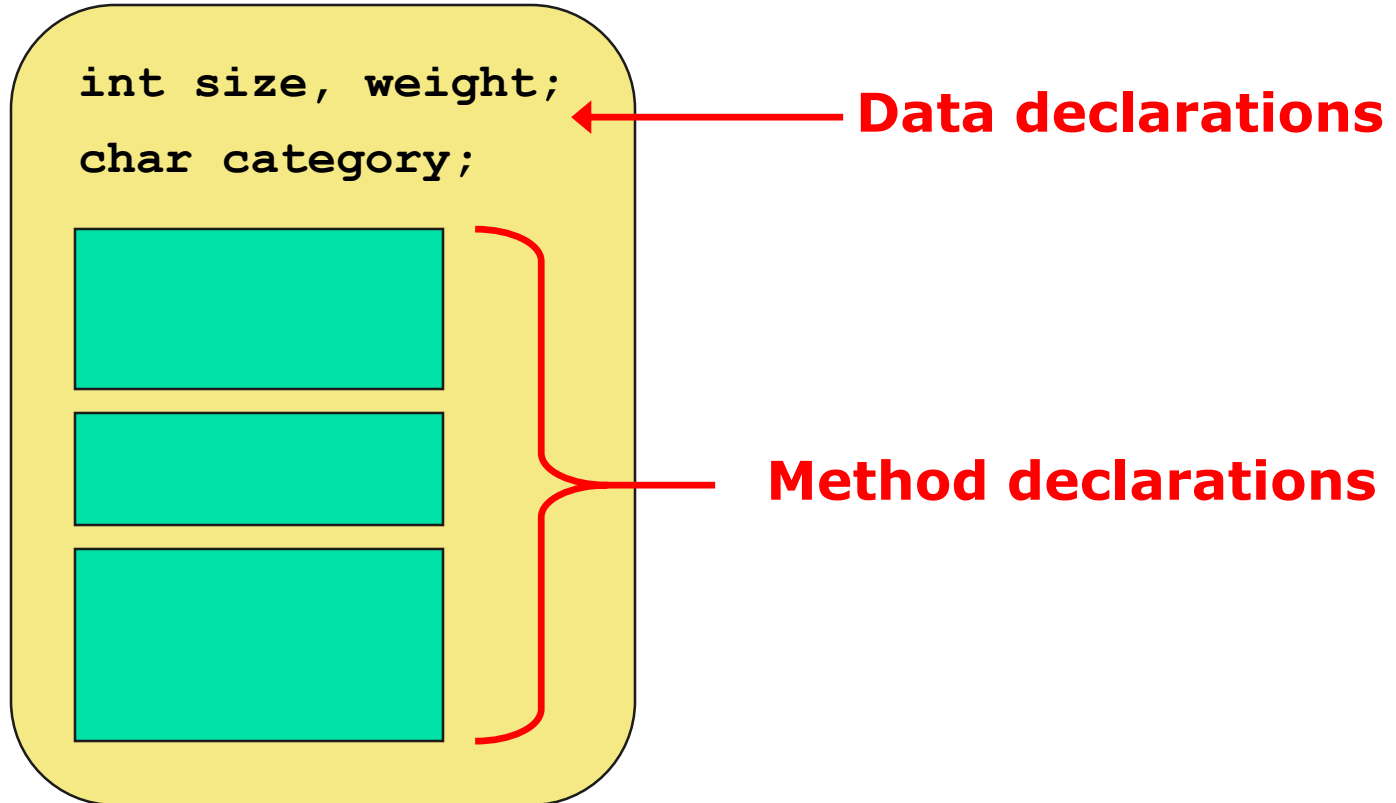
Deposit()

Withdraw()

checkBalance()

Classes

- A class can contain data declarations and method declarations



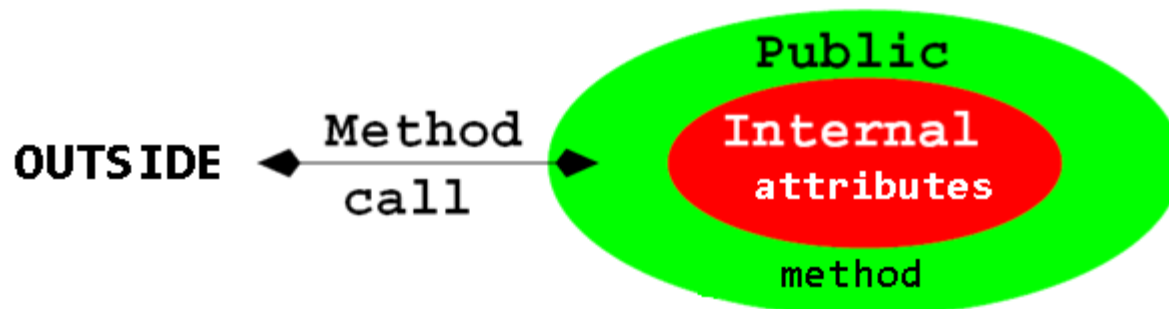


Phạm vi truy cập

- Phạm vi truy cập xác định khả năng nhìn thấy được của một thành phần của chương trình với các thành phần khác của chương trình
- Đối với lớp
 - Phạm vi truy cập có thể được áp dụng cho các thành phần của lớp
 - **private**: chỉ truy cập được từ bên trong lớp đó
 - **public**: có thể truy cập được tại mọi nơi, từ trong và ngoài lớp

Che giấu dữ liệu

- Sử dụng phạm vi truy cập để che giấu dữ liệu: tránh thay đổi trái phép hoặc làm sai lệch dữ liệu
- Dữ liệu được che giấu ở bên trong lớp bằng cách gán phạm vi truy cập *private*.
 - chỉ có thể truy cập từ các phương thức bên trong lớp
- Các đối tượng khác muốn truy nhập vào dữ liệu riêng tư này phải thông qua các phương thức của lớp có phạm vi truy cập *public*.





Che giấu dữ liệu (2)

- Để truy cập và chỉnh sửa các giá trị của dữ liệu, lớp cần phải cung cấp các dịch vụ
 - Accessor (getter): Trả về giá trị hiện tại của một thuộc tính (dữ liệu)
 - Mutator (setter): Thay đổi giá trị của một thuộc tính
 - Thường là getX và setX, trong đó X là tên thuộc tính



Ví dụ: phương thức get, set

```
class Student{  
    private String name;  
    public String getName() {  
        return this.name;  
    }  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```




Khai báo lớp

- Cú pháp: sử dụng từ khóa **class**

```
class Ten_Lop {  
    // Nội dung lớp  
    // Khai báo các thuộc tính  
    // Khai báo và cài đặt các phương thức  
}
```

Ví dụ

```
class Student {  
    // Nội dung lớp  
}
```

Khai báo Thuộc tính

- Cú pháp khai báo thuộc tính: Tương tự khai báo biến

`phamViTruyCap kieu tenThuocTinh;`

- Thuộc tính có thể được khởi tạo khi khai báo
 - Các giá trị mặc định sẽ được sử dụng nếu không được khởi tạo.

- Ví dụ

access modifier

type

```
package com.megabank.models;  
  
public class BankAccount {  
    private String owner;  
    private double balance = 0.0;  
}
```

name

Khai báo Phương thức

- Khai báo: tương tự khai báo hàm
- Cú pháp

```
phamViTruyCap kiểuTrảVề tênPhươngThức (ds  
    tham số) {  
    // Nội dung phương thức  
}
```

- Ví dụ

The diagram shows a Java method declaration: `public boolean debit(double amount) {`. Four labels with arrows point to specific parts of the declaration: 'access modifier' points to 'public', 'return type' points to 'boolean', 'method name' points to 'debit', and 'parameter list' points to '(double amount)'. The rest of the method body is shown as comments: `// Method body` and `// Java code that implements method behavior`.

```
public boolean debit(double amount) {  
    // Method body  
    // Java code that implements method behavior  
}
```

Khai báo Phương thức

- Mỗi phương thức phải có một chữ ký riêng, phân biệt các phương thức, gồm:
 - Tên phương thức
 - Số lượng các tham số và kiểu của chúng

method name argument type

```
public void credit(double amount) {  
    ...  
}
```

signature



Khai báo Phương thức

- Khi phương thức trả về ít nhất một giá trị hoặc một đối tượng thì bắt buộc phải có câu lệnh `return` để trả điều khiển cho đối tượng gọi phương thức.
- Nếu phương thức không trả về 1 giá trị nào (`void`) không cần câu lệnh `return`
- Có thể có nhiều lệnh `return` trong một phương thức; câu lệnh đầu tiên mà chương trình gặp sẽ được thực thi.

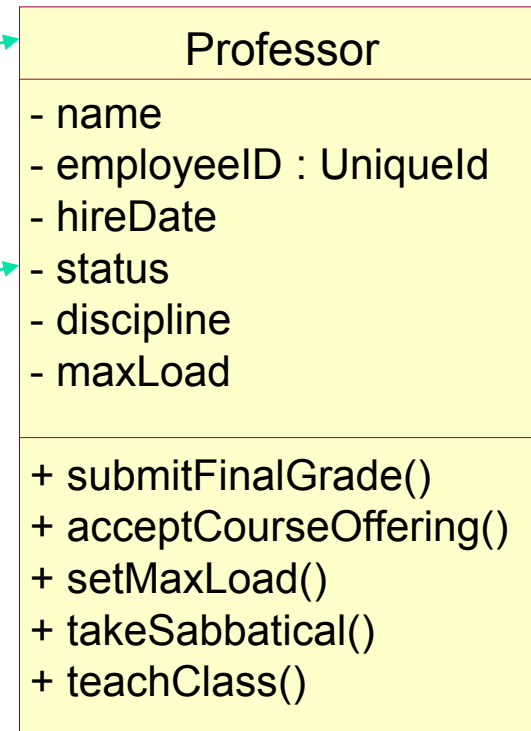
Biểu diễn trong UML

- Lớp (class) được biểu diễn bằng 1 hình chữ nhật với 3 thành phần:

- Tên lớp

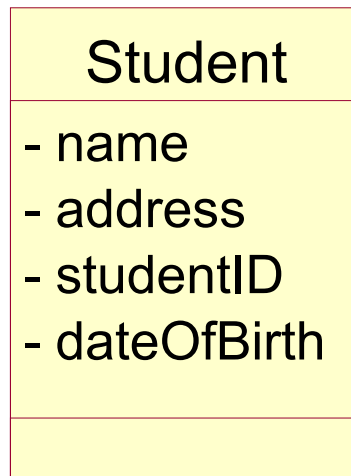
- Thuộc tính

- Phương thức

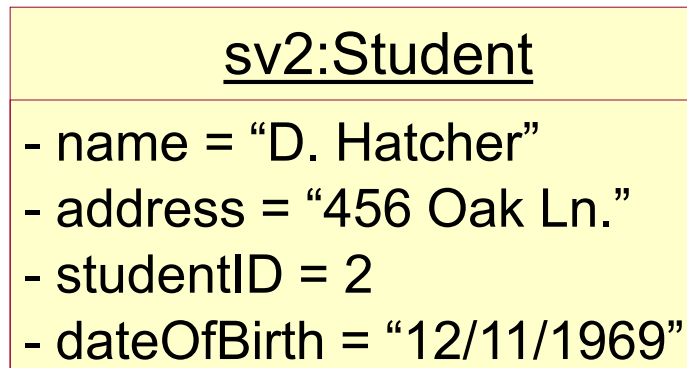
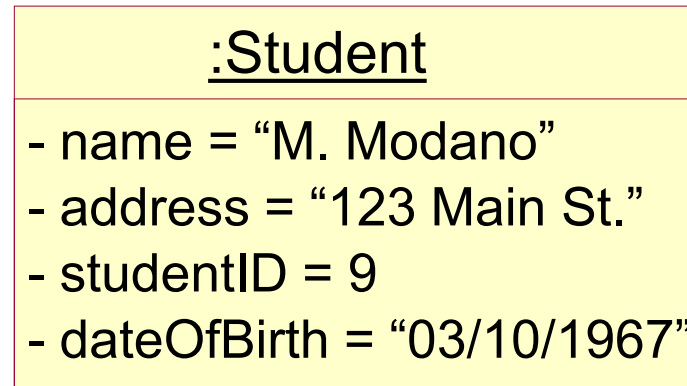


Biểu diễn trong UML (2)


- Đối tượng: biểu diễn bằng *tên đối tượng:tên lớp*, và các giá trị của thuộc tính.



Class



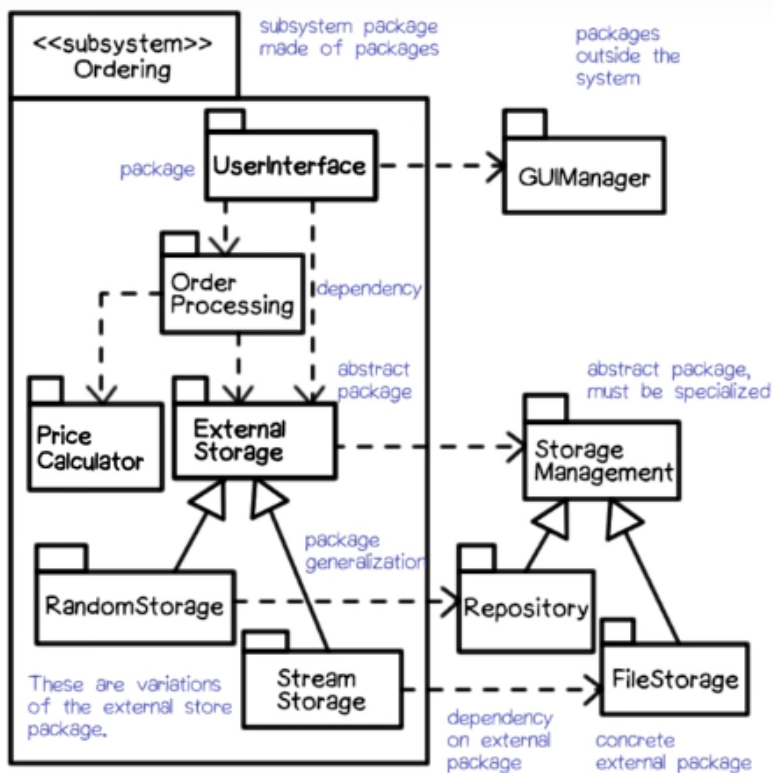
Objects



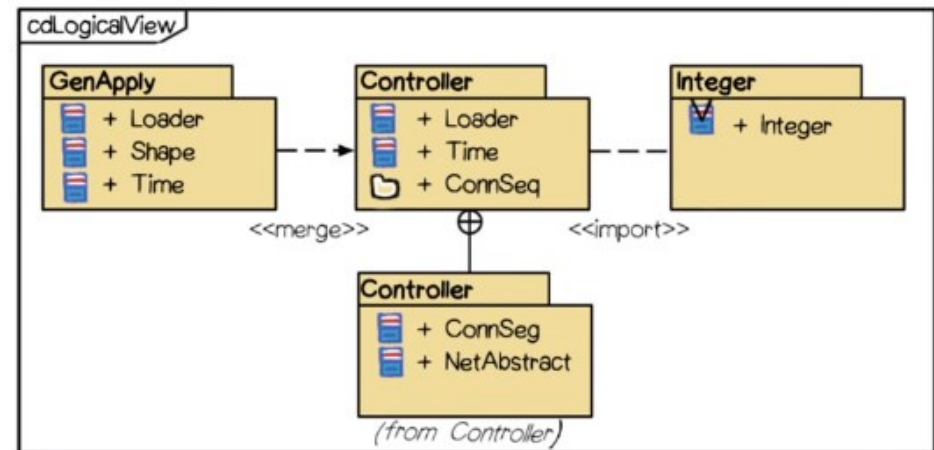
Biểu diễn trong UML (3)

■ Biểu diễn gói trong UML

UML 1.5



UML 2.0





Khai báo gói

- Cú pháp khai báo:

```
package tenpackage;  
chi_dinh_truy_cap class TenLop {  
    // Than lop  
}
```

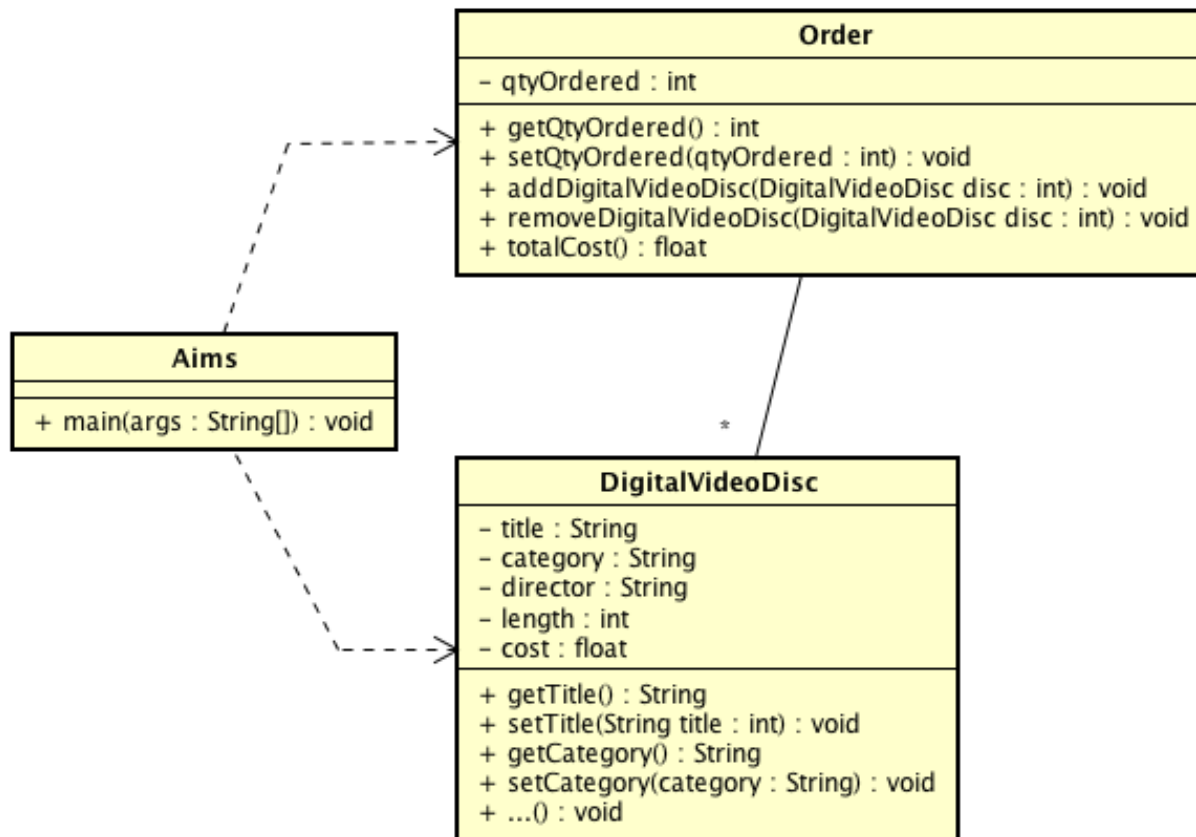
- `chi_dinh_truy_cap`:

- public: Lớp có thể được truy cập từ bất cứ đâu, kể cả bên ngoài package chứa lớp đó.
- Không chỉ định: Lớp chỉ có thể được truy cập từ bên trong package chứa lớp đó.

```
package oop.k52.cnpm;  
    public class Student {  
        ...  
    }
```

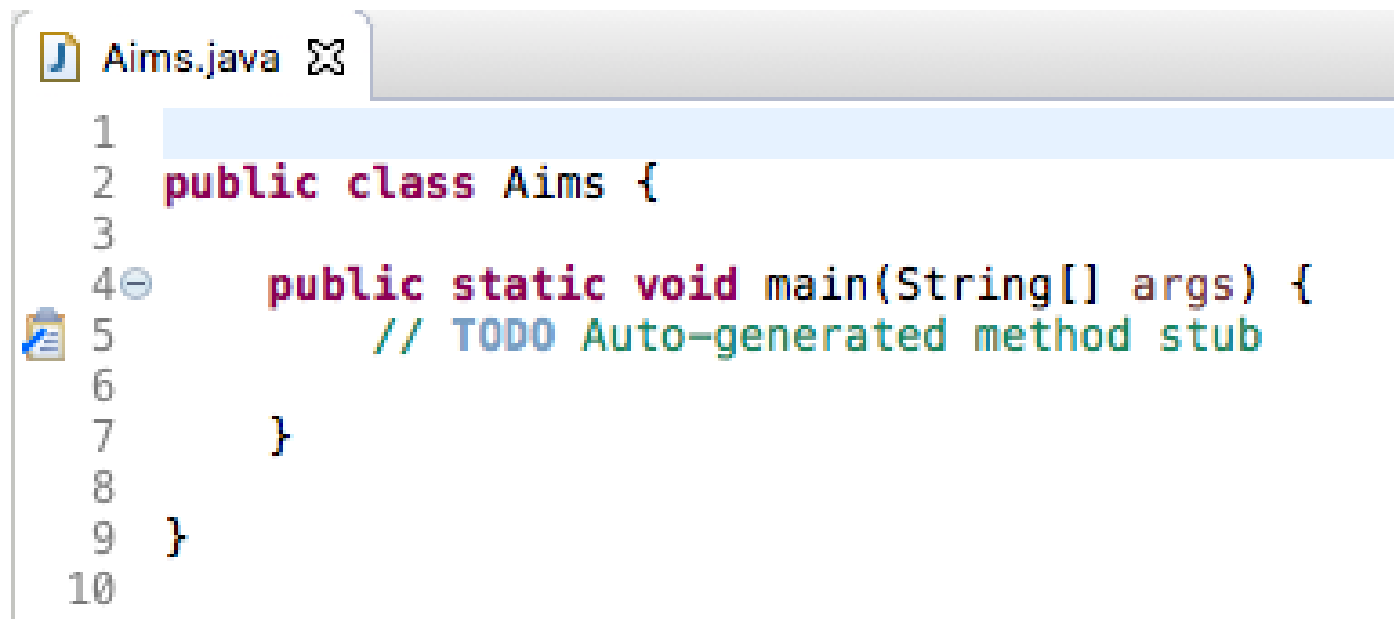
Bài 1

■ UML Class Diagram of the system



Bài 1

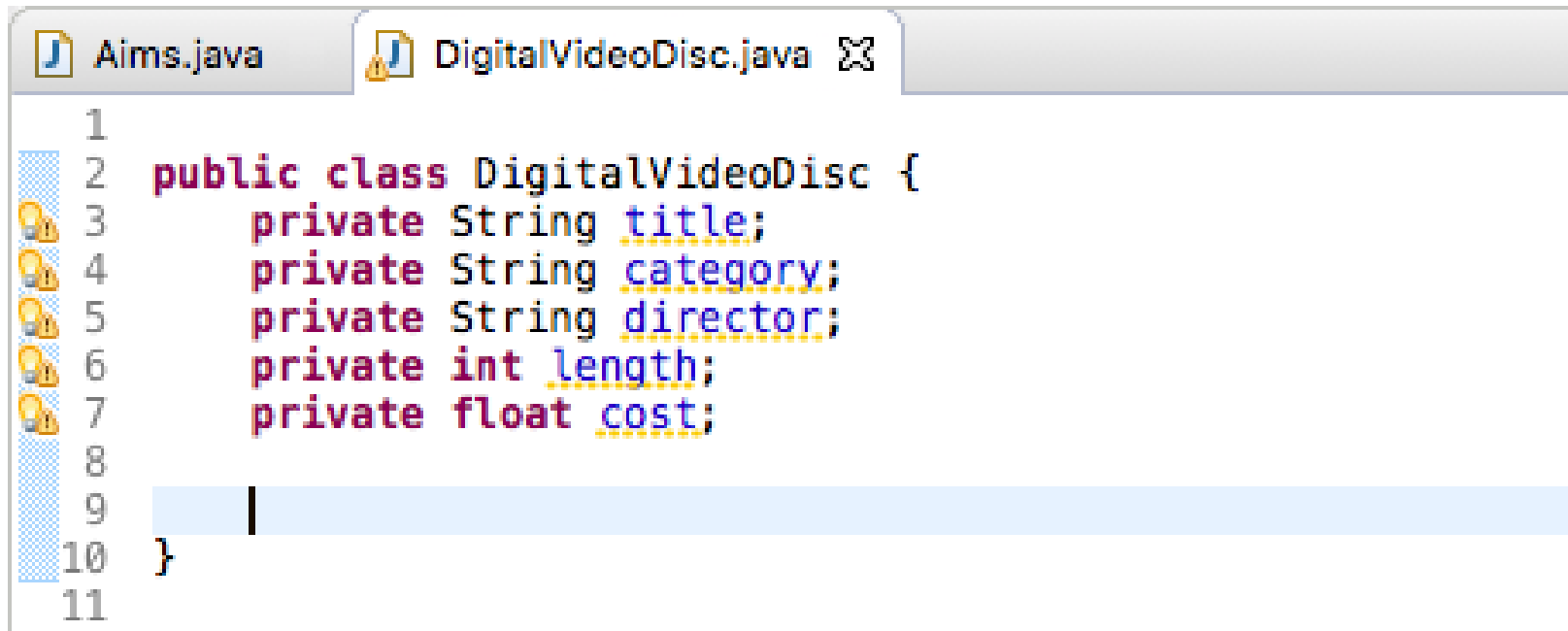
- 1. Create Aims class
 - Open Eclipse
 - Create a new JavaProject named "AimsProject"
 - Create Aims class



```
1  
2 public class Aims {  
3  
4     public static void main(String[] args) {  
5         // TODO Auto-generated method stub  
6  
7     }  
8  
9 }  
10
```

Bài 1

- 2. Create the DigitalVideoDisc class and its attributes
 - Open the source code of the DigitalVideoDisc class and add some attributes below



```
1
2 public class DigitalVideoDisc {
3     private String title;
4     private String category;
5     private String director;
6     private int length;
7     private float cost;
8
9
10 }
11
```

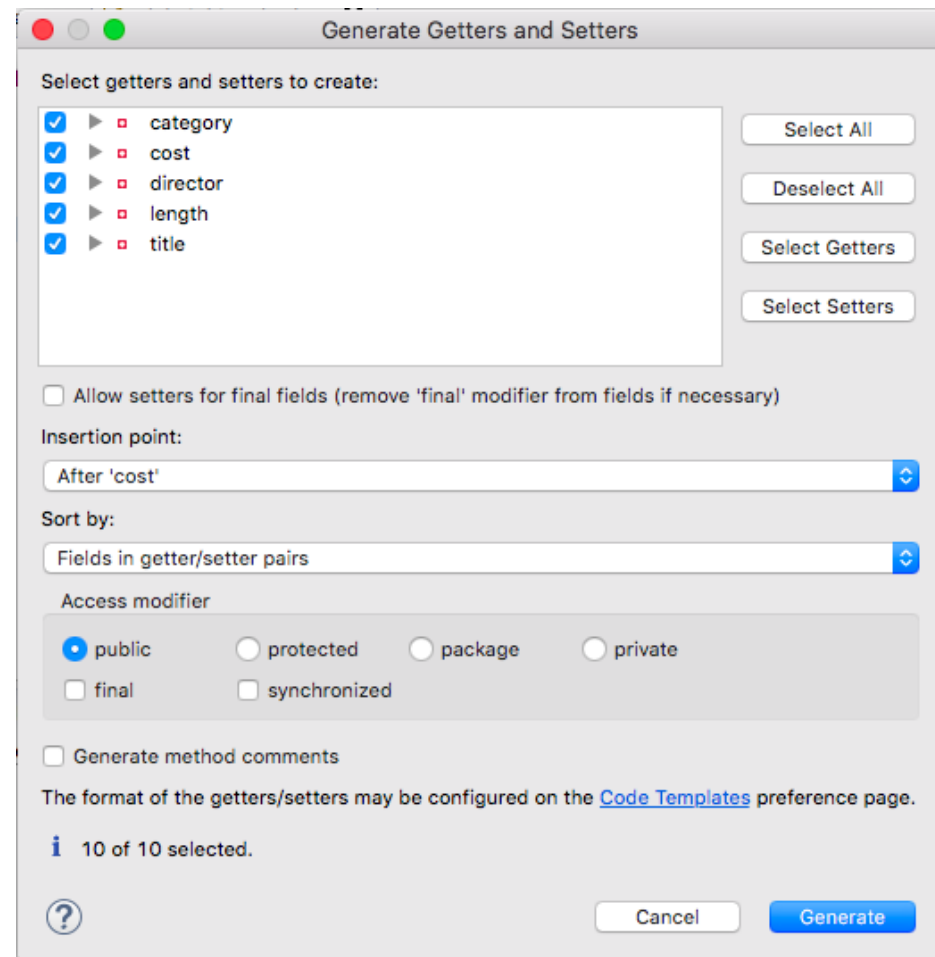
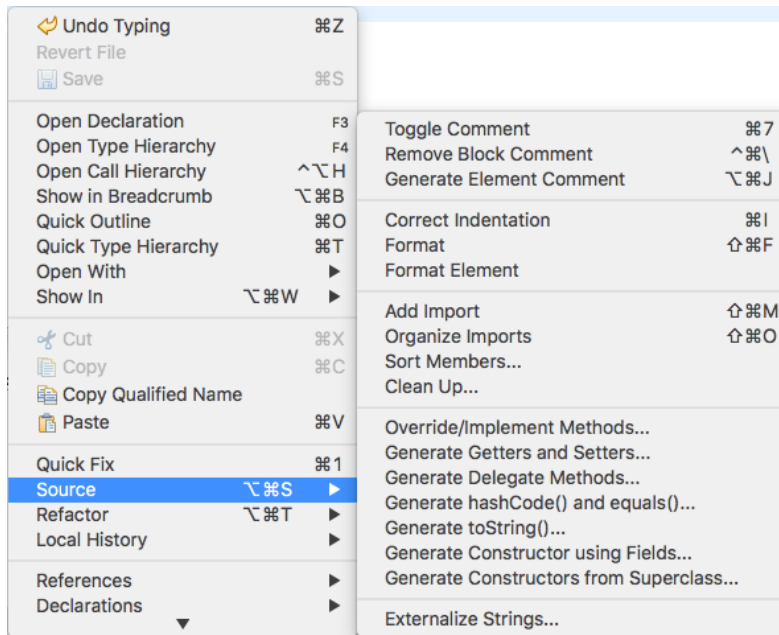


Bài 1

- 3. Create accessors and mutators for the class DigitalVideoDisc
 - Right click anywhere in the source file of DigitalVideoDisc.
 - Choose Source, then choose Generate Getters and Setters
 - Choose all the attributes
 - Choose the option "public" in the Access modifier
 - Click Generate

Bài 1

■ 3. Create accessors and mutators for the class DigitalVideoDisc





Bài 1

- 4. Create Constructor method
 - In this part, you will create yourself constructor method for DigitalVideoDisc for different purposes:
 - Create a DVD object by title
 - Create a DVD object by category and title
 - Create a DVD object by director, category and title
 - Craete a DVD object by all attributes: title, category, director, length and cost



Bài 1

- 4. Create Constructor method
 - Eclipse also allows you to generate automatically constructor methods by field.
 - Just do the same as generating getters and setters. Right click any where in the source file, Choose Source, Choose Generate constructors by fields then select fields to generate constructor methods.



Bài 1

- 5. Create the Order class to work with DigitalVideoDisc
 - The Order class will contain a list of DigitalVideoDisc objects and have methods capable of modifying the list.
 - In this class, you will create a constant to indicate the maximum number of items that a customer can buy. Supposing that one can buy maximum 10 items.
 - Then, you add a field as an array to store a list of DigitalVideoDisc.

```
public class Order {  
  
    public static final int MAX_NUMBERS_ORDERED = 10;  
    private DigitalVideoDisc itemsOrdered[] = new DigitalVideoDisc[MAX_NUMBERS_ORDERED];  
  
}
```



Bài 1

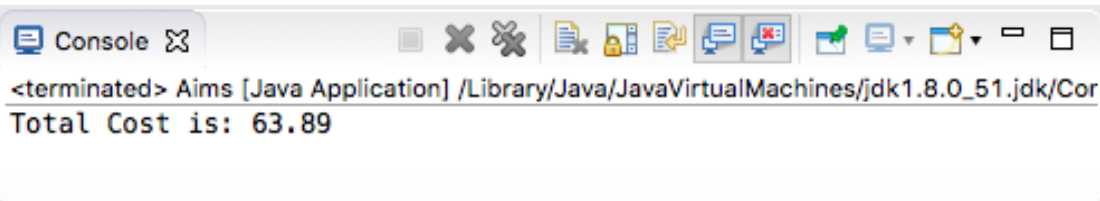
- 5. Create the Order class to work with DigitalVideoDisc
 - To keep track of how many DigitalVideoDiscs are in the order, you must create a field named **qtyOrdered** in the Order class which stores this information.
 - Generate getter and setter methods for the qtyOrdered field (not for the **itemsOrdered** field)
 - Create the method **addDigitalVideoDisc(DigitalVideoDisc disc)** to add more an item to the list. You should check the current number of quantity to assure that the order is not already full
 - Create the method **removeDigitalVideoDisc(DigitalVideoDisc disc)** to remove the item passed by argument from the list.
 - Create the **totalCost()** method which loops through the values of the array and sums the costs of the individual DigitalVideoDiscs. This method returns the total cost of the current order.

Bài 1

- 6. Create Orders of DigitalVideoDiscs
 - The Aims class should create a new Order, and then create new DVDs and populate the order with those DVDs. This will be done in the main() method of the Aims class.
 - Do the following code in your main method and run the program to test.

The result should be:

```
public static void main(String[] args) {  
    // TODO Auto-generated method stub  
    Order anOrder = new Order();  
    // Create a new dvd object and set the fields  
    DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King");  
    dvd1.setCategory ("Animation");  
    dvd1.setCost (19.95f);  
    dvd1.setDirector ("Roger Allers");  
    dvd1.setLength (87);  
    // add the dvd to the order  
    anOrder.addDigitalVideoDisc(dvd1);  
  
    DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star Wars");  
    dvd2.setCategory ("Science Fiction");  
    dvd2.setCost (24.95f);  
    dvd2.setDirector ("George Lucas");  
    dvd2.setLength (124);  
    anOrder.addDigitalVideoDisc(dvd2);  
  
    DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladdin");  
    dvd3.setCategory ("Animation");  
    dvd3.setCost (18.99f);  
    dvd3.setDirector ("John Musker");  
    dvd3.setLength (90);  
  
    // add the dvd to the order  
    anOrder.addDigitalVideoDisc(dvd3);  
  
    System.out.print ("Total Cost is: ");  
    System.out.println (anOrder.totalCost());  
}
```





Bài 1

- **7. You have to write code to test the remove function of the Order class and check if the code is successfully run.**



Bài 2

- 8. Tạo một lớp **MyDate** gồm 3 thuộc tính ngày, tháng, năm (date, month, year) – số nguyên dương
 - Viết các phương thức set/get cho các thuộc tính của lớp **MyDate**.
 - Xây dựng 2 phương thức khởi tạo:
 - Một phương thức không có tham số
 - Một phương thức có 3 tham số có kiểu là kiểu của 3 thuộc tính của lớp.
 - Tạo phương thức nhập các thuộc tính cho đối tượng MyDate từ bàn phím và phương thức in ra ngày, tháng, năm của đối tượng MyDate ra màn hình.
 - Viết hàm main thực hiện các lệnh để kiểm tra các phương thức của lớp đã xây dựng



Bài 3

- 9. Tạo một lớp số phức (**Complex**) gồm các thuộc tính: Phần thực và Phần ảo (số thực)
 - Viết các phương thức get/set cho các thuộc tính của lớp Complex nhằm đảm bảo tính đóng gói.
 - Viết các phương thức khởi tạo sau đây cho lớp **Complex**:
 - Không có tham số nào: gán phần thực và phần ảo = 0
 - Nhận hai số thực làm tham số
 - Viết phương thức nhập vào một số phức từ bàn phím với chữ ký như sau:
 - `public void nhapSoPhuc(Complex cmp)`
 - Viết các phương thức cộng, trừ, nhân, chia hai số phức với các chữ ký như sau:
 - `public Complex congHaiSoPhuc(Complex cmp1, Complex cmp2)`
 - `public Complex truHaiSoPhuc(Complex cmp1, Complex cmp2)`
 - `public Complex nhanHaiSoPhuc(Complex cmp1, Complex cmp2)`
 - `public Complex chiaHaiSoPhuc(Complex cmp1, Complex cmp2)`
 - Viết phương thức in thông tin một số phức ra màn hình
 - `public inSoPhuc(Complex cmp)`
 - Viết hàm main kiểm tra các phương thức đã cài đặt cho lớp