บทพิเศษ

OOP Programming with Java

หัวข้อที่จะเรียนรู้

- แนวคิด Object-Oriented Programming (OOP)
- Class และ Object
- Encapsulation & Abstraction
- Inheritance & Polymorphism
- การประยุกต์ใช้ OOP ใน Java

OOP คืออะไร?

- Paradigm การเขียนโปรแกรมที่ใช้ **Object** เป็นศูนย์กลาง
- Object = ข้อมูล (Attributes) + พฤติกรรม (Methods)
- ช่วยให้โค้ด:
  - เข้าใจง่าย
  - Reusable
  - Extensible
  - Maintainable

4 หลักการ OOP

1. **Encapsulation** – ซ่อนข้อมูลภายใน คลาสควบคุมการเข้าถึง
2. **Abstraction** – ซ่อนรายละเอียดที่ไม่จำเป็น เหลือเฉพาะสิ่งสำคัญ
3. **Inheritance** – สืบทอดคุณสมบัติ/พฤติกรรมจากคลาสแม่
4. **Polymorphism** – ใช้ method เดียวกันแต่ทำงานต่างกันตามชนิด object

Class & Object

```
public class Student {
    String name;
    int age;

    public void study() {
        System.out.println(name + " is studying.");
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.name = "Alice";
        s1.age = 20;
        s1.study();
    }
}
```

Encapsulation

- ใช้ **private fields + getter/setter methods**
- ป้องกันการเข้าถึงข้อมูลโดยตรง

```
public class BankAccount {
    private double balance;

    public void deposit(double amount) {
        balance += amount;
    }

    public double getBalance() {
        return balance;
    }
}
```

Inheritance

```
class Person {
    String name;
    void introduce() {
        System.out.println("Hi, I'm " + name);
    }
}

class Student extends Person {
    String studentId;
}
```

Polymorphism

```
class Animal {
    void sound() { System.out.println("Some sound"); }
}

class Dog extends Animal {
    void sound() { System.out.println("Woof!"); }
}

class Cat extends Animal {
    void sound() { System.out.println("Meow!"); }
}
```

```
Animal a1 = new Dog();
Animal a2 = new Cat();
a1.sound(); // Woof!
a2.sound(); // Meow!
```

Lab (วันนี้)

1. เขียน Class `Person` (name, age, introduce())
2. เขียน Subclass `Student` (studentId) และ `Teacher` (salary)
3. เขียน Interface `Course` (getCourseName()) และให้ `Student` implement
4. ทดลอง Polymorphism ระหว่าง Student/Teacher

Assignment

- พัฒนา Mini Project: **Student Management System**
  - Class `Person` (name, age)
  - Subclass `Student` (studentId, GPA)
  - Subclass `Teacher` (salary, subject)
  - ใช้ Polymorphism แสดงข้อมูล `introduce()`
- ส่ง: Source code + Screenshot output