บทที่ 3
CLASSIFICATION, GENERALIZATION AND SPECIALIZATION

อ.สกรณ์ บุษบง
สาขาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์
มหาวิทยาลัยราชภัฏบุรีรัมย์

OVERVIEW

- Object ที่มีคำจำกัดความที่คล้ายกัน จะถูดจัดให้อยู่ใน class เดียวกัน
- การระบุและจัดหมวดหมู่ของ object ที่มีความคล้ายกันให้อยู่ class เดียวกัน เรียกว่า Classification

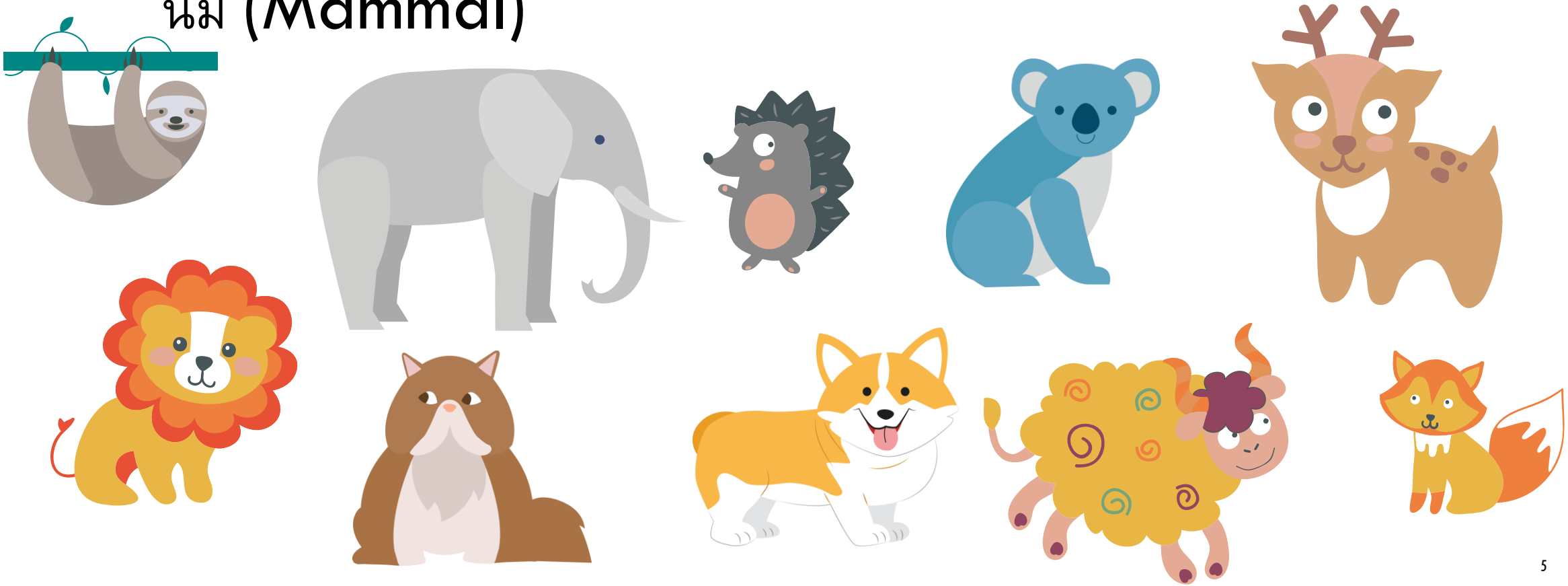
OVERVIEW

- ในบทนี้เราจะดูว่า
- **object** ถูกจัดกลุ่มจาก **class** ยังไง
- **Relationship** สามารถจัดการ **class** ให้อยู่ใน **class hierarchy** โดยใช้ **generalization** และ **specialization** ยังไง

CLASSIFICATION

CLASSIFICATION

- ลองจำแนกประเภทของสัตว์ พบว่า สัตว์เหล่านี้เป็นสัตว์เลี้ยงลูกด้วยนม (Mammal)

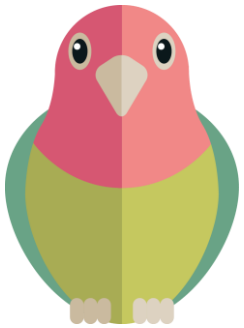


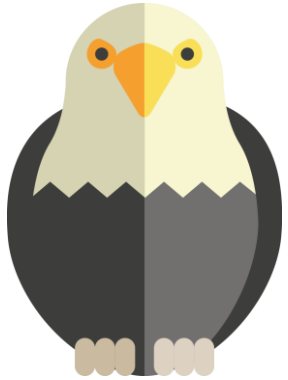
CLASSIFICATION

- สัตว์ที่อยู่กลุ่มของ **mammal** จะมีคุณสมบัติของสัตว์เลือดอุ่นต่างๆที่เหมือนๆกัน เช่น
  - เป็นสัตว์เลือดอุ่น
  - หายใจโดยใช้ปอด
  - มีขนปกคลุมร่างกาย

CLASSIFICATION

- ลองจำแนกสัตว์ที่อยู่ในกลุ่มนก (Bird)



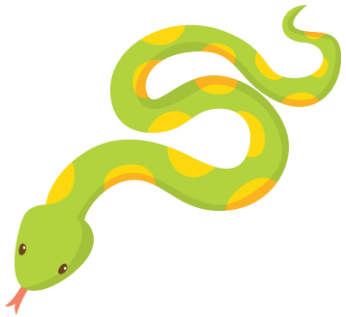


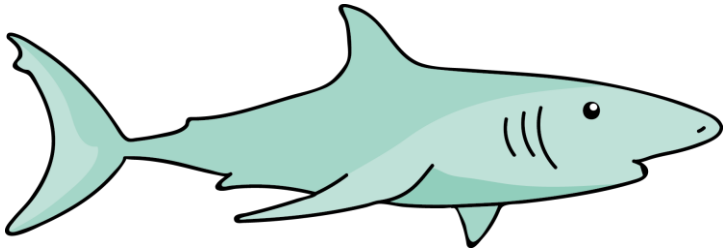
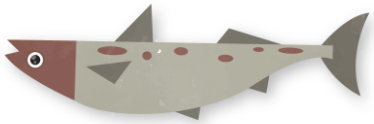
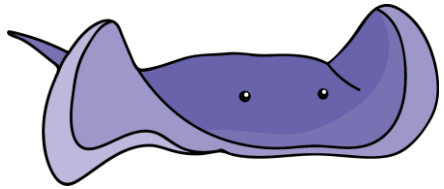
CLASSIFICATION

- สัตว์ที่อยู่กลุ่มของ bird จะมีคุณสมบัติของสัตว์ปีก
  - มีจะงอยปาก
  - มีขา 2 ข้าง
  - มีปีก 2 ข้าง
  - ปีกปกคลุมไปด้วย feathers
  - บินได้
  - ออกลูกเป็นไข่
  - เป็นสัตว์เลือดอุ่น

CLASSIFICATION

- ลองจัดกลุ่มอื่นๆดู


CLASSIFICATION

- การที่เรานำสัตว์ต่าง ๆมาแบ่งกลุ่มแบบนี้ เราใช้ลักษณะที่สัตว์มีคล้าย ๆกันให้อยู่กลุ่มเดียวกัน
- เราเรียนกระบวนการนี้ว่า **classification**
- กลุ่มต่าง ๆของสัตว์เราเรียกว่า **class**

HIERARCHICAL RELATIONSHIP OF CLASSES

- Class ต่างๆสามารถจัดการได้โดยการใช้ลำดับชั้น (Hierarchical)
- จริงๆแล้ว Class ต่างๆ จะมีตำแหน่งที่เหมาะสมอยู่ในลำดับชั้น (Hierarchical) ซึ่งมี 2 ลักษณะคือ Superclass และ Subclass
- นอกจากนั้นการจำลองลำดับชั้น (Hierarchical) ใช้ Class Hierarchical Diagram ในการแบ่งลำดับชั้น

HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS

- ในบทที่ 2 ได้แนะนำให้รู้จัก **Bill** กับ **Sarah** แล้ว ในบทนั้นได้พบกับพนักงานขาย 2 คน คือ

Jules

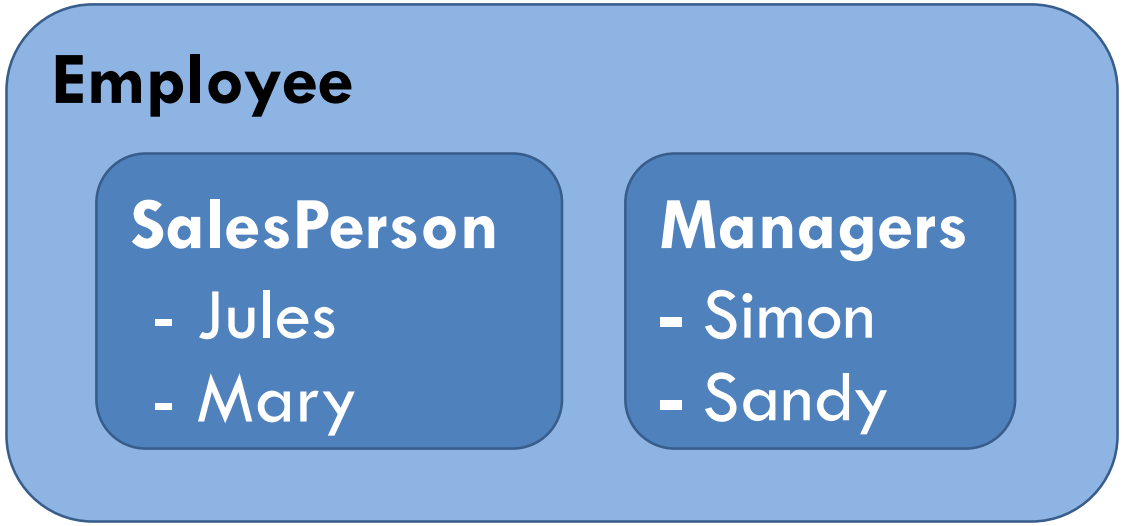
Mary

HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS

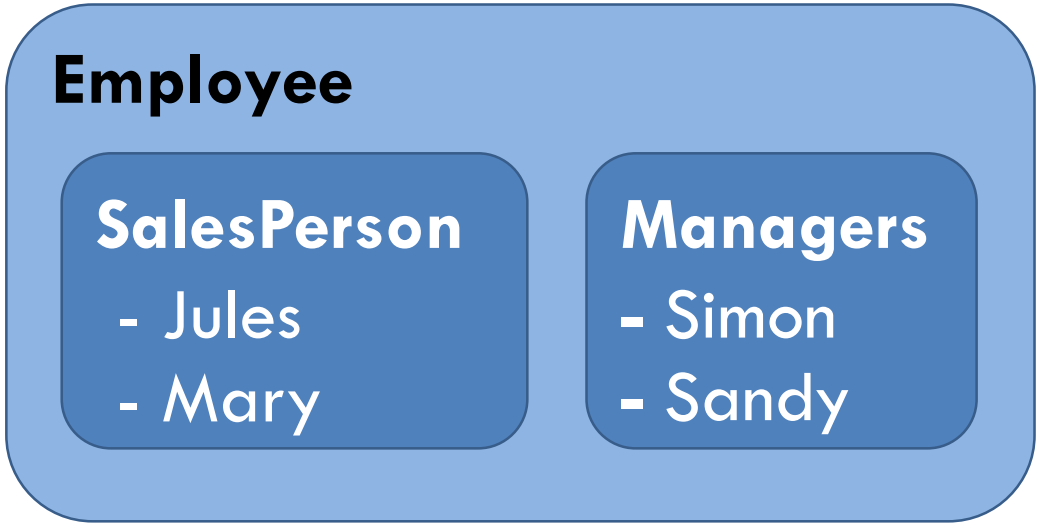
- ใบบทนี้ขอแนะนำ Simon และ Sandy
- Simon และ Sandy ไม่ใช่พนักงานขาย แต่มีตำแหน่งเป็นผู้จัดการ
- ผู้จัดการ (Managers) มีคุณสมบัติที่แตกต่างกับพนักงานขาย (Salesperson) นิดหน่อย
- ดังนั้น Simon และ Sandy จัดได้ว่าเป็น object ที่มาจาก class ที่ต่างจาก Jules และ Mary ที่มีตำแหน่ง พนักงานขาย
- ทั้งผู้จัดการ และพนักงานขาย ต่างก็เป็น พนักงาน(Employee) เหมือนกัน

HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS

- ดังนั้นความสัมพันธ์ที่เกิดขึ้นจะมีลักษณะดังนี้



HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS



- จะเห็นว่า Jules และ Mary เป็น SalesPerson ในขณะเดียวกันก็เป็น Employee ด้วย
- Simon และ Sandy เป็น Managers แต่ก็เป็น Employee เช่นเดียวกัน

HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS

- ดังนั้น Jules Mary Simon และ Sandy ต่างก็เป็นพนักงาน (Employee) ทำให้ทุกคนมีสภาพทั่วไป (generality) ของพนักงานที่ควรมี
- แต่ Jules และ Mary มีตำแหน่งเป็นพนักงานขาย (SalesPerson) จึงมีสภาพที่เฉพาะเจาะจง (specifically) ในขณะเดียวกันก็ยังคงมีสภาพทั่วไป (generality) ของพนักงานด้วย

HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS

- เช่น เมื่อ Mary ได้จัดการรายการสั่งซื้อของลูกค้าเสร็จสิ้น เธอจะได้รับค่า **commission** จากการขายแต่ละครั้ง ลักษณะแบบนี้จะไม่เกิดขึ้นกับ Sandy ที่มีตำแหน่งเป็น Manager แม้ว่าทั้งคู่จะเป็นพนักงานที่ร้านจำหน่ายรถยนต์เช่นกัน

HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS

- class Employee เป็น *generalized class* ของ class SalesPerson และ class Manager
- ในทางกลับกัน class SalesPerson และ class Manager ก็เป็น *specialized class* ของ class Employee

HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS

- Generalized และ specialized classes สามารถเขียนใส่ใน class hierarchy ได้โดย
  - generalized classes placed toward the top of the hierarchy
  - specialized classes toward the bottom of the hierarchy

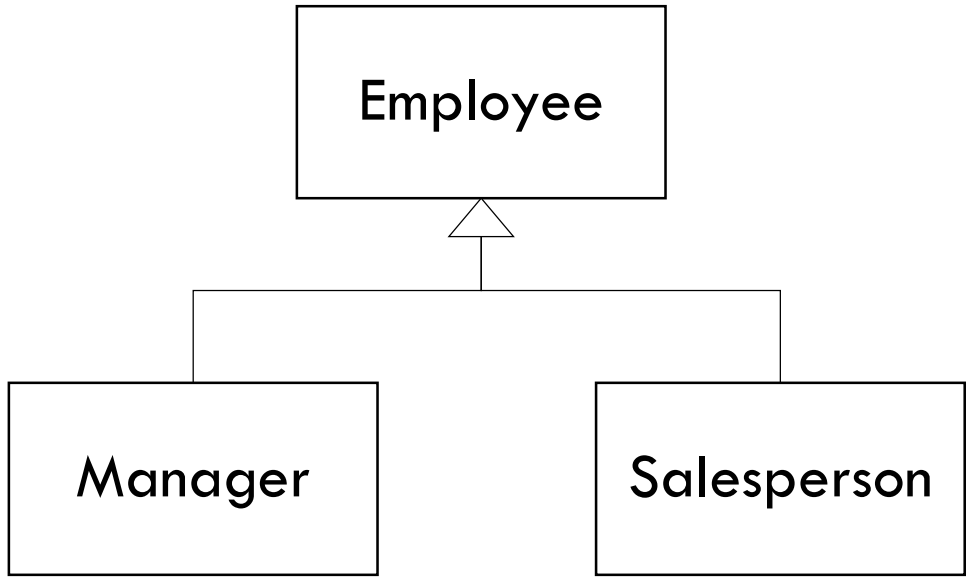
HIERARCHICAL RELATIONSHIP OF CLASSES >SUPERCLASS AND SUBCLASS

- *specialized class* คือ subclass
- *generalized class* คือ superclass
- ดังนั้น SalesPerson จึงเป็น subclass ของ Employee
- ในขณะเดียวกัน Employee ก็เป็น superclass ของ SalesPerson

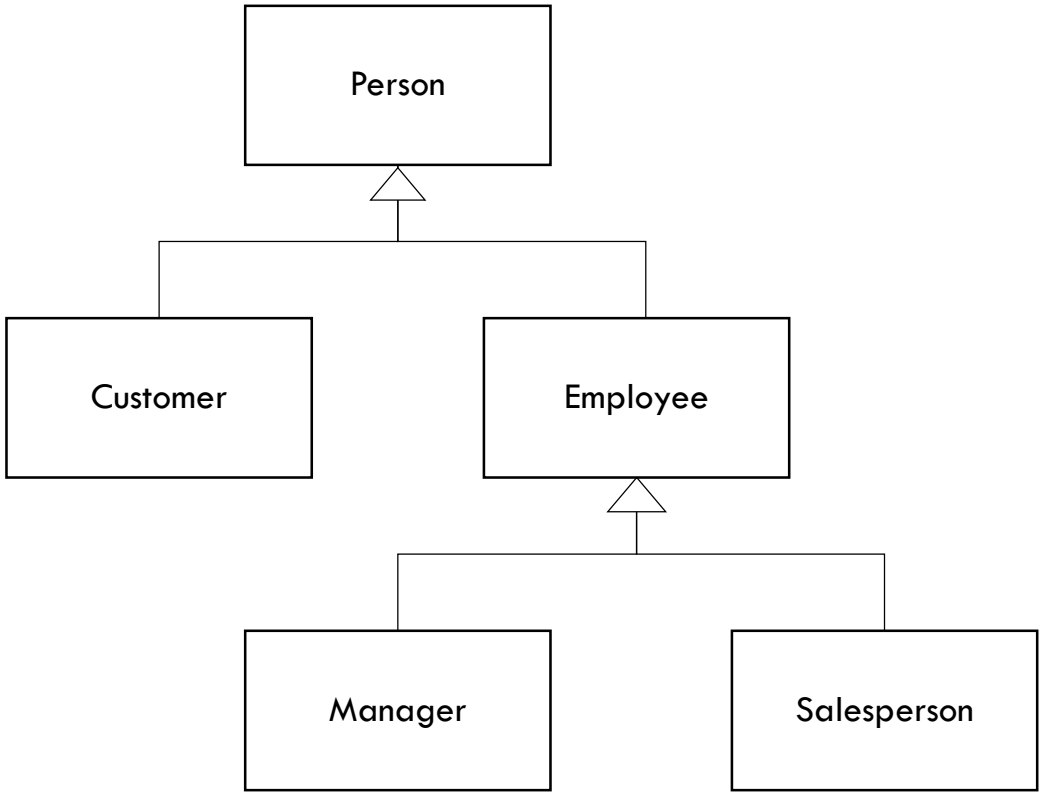
CLASS HIERARCHY DIAGRAM

- ในการแสดงลำดับความสัมพันธ์สามารถใช้ class hierarchy diagram เพื่อแสดงความสัมพันธ์ได้
- ดังนั้นความสัมพันธ์ระหว่าง Employee Salesperson และ Manager จะมี class hierarchy diagram ดังนี้

CLASS HIERARCHY DIAGRAM



CLASS HIERARCHY DIAGRAM



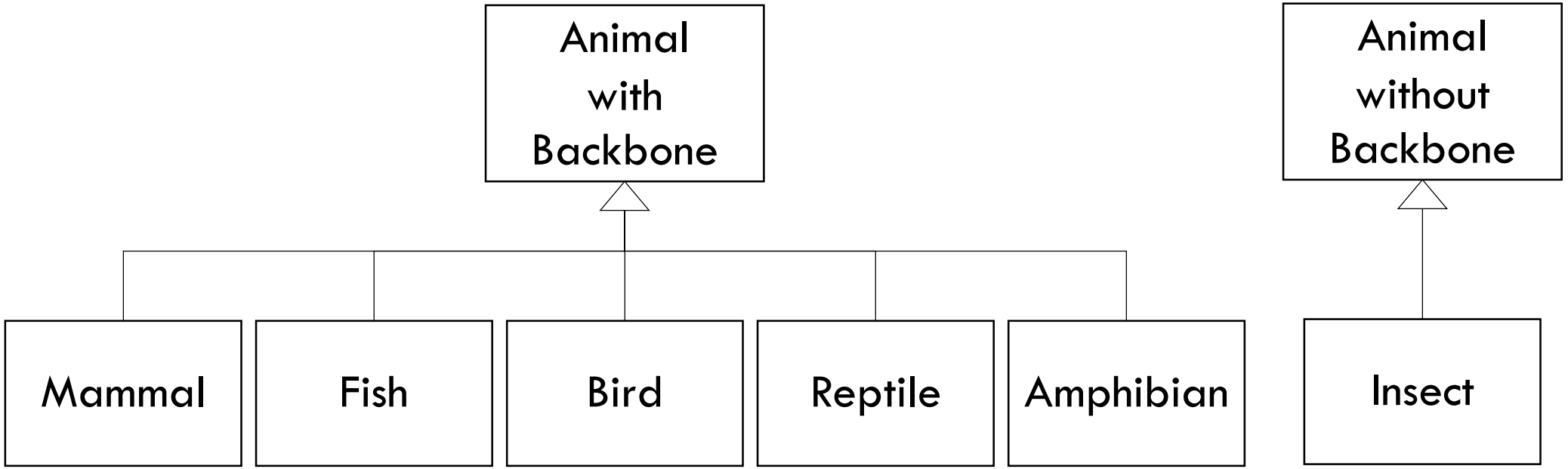
GENERALIZATION

- **Generalization** คือการตรวจจับการคล้ายคลึงระหว่าง **class** จากนั้นกำหนดคล้ายคลึงนั้นเป็น *generalized class* ใหม่
- จากนั้น **class** อื่นๆที่ถูกตรวจจับจะหลายเป็น **subclass** ของ *generalized class*
- เช่น **class Mammal, Fish, Bird, Reptile,** และ **Amphibian** ต่างก็มีความคล้ายคลึงกันอยู่ ซึ่งก็คือ สัตว์ที่มีกระดูกสันหลัง

GENERALIZATION

- จากความคล้ายคลึงของสัตว์ต่างๆ เหล่านี้เราสามารถสร้าง **superclass** ใหม่ได้ คือ **class Animal-with-Backbone**
- ในขณะเดียวกัน **class Insect** สามารถ **generalized** เป็น **class Animal-without-Backbone**

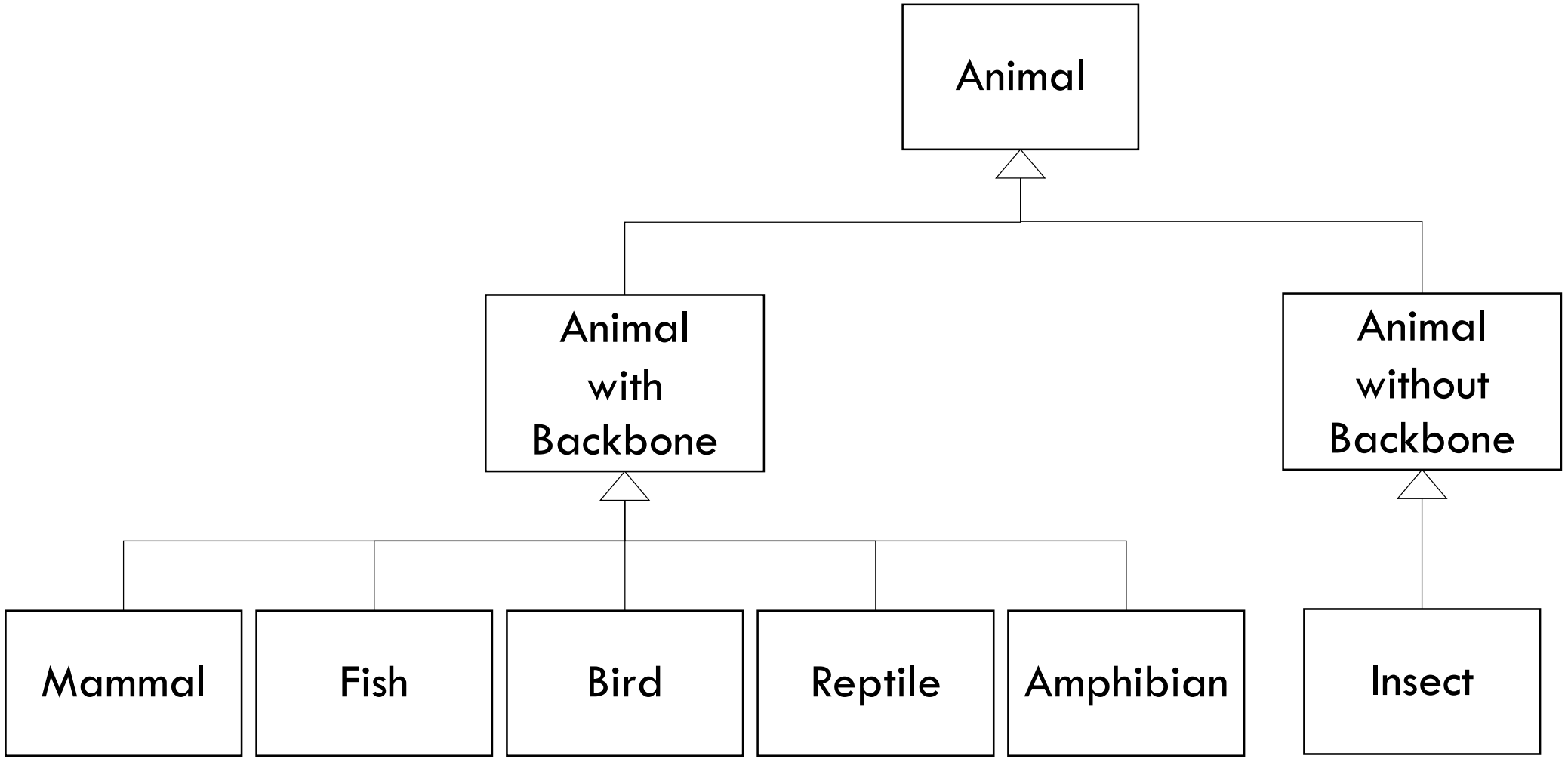
GENERALIZATION



GENERALIZATION

- Class Animal-with-Backbone และ class Animal-without-Backbone สามารถทำการ generalized ต่อได้อีกโดยพิจารณาความคล้ายคลึงกันระหว่าง 2 class นี้
- จะได้ *generalized class* ใหม่ คือ Animal

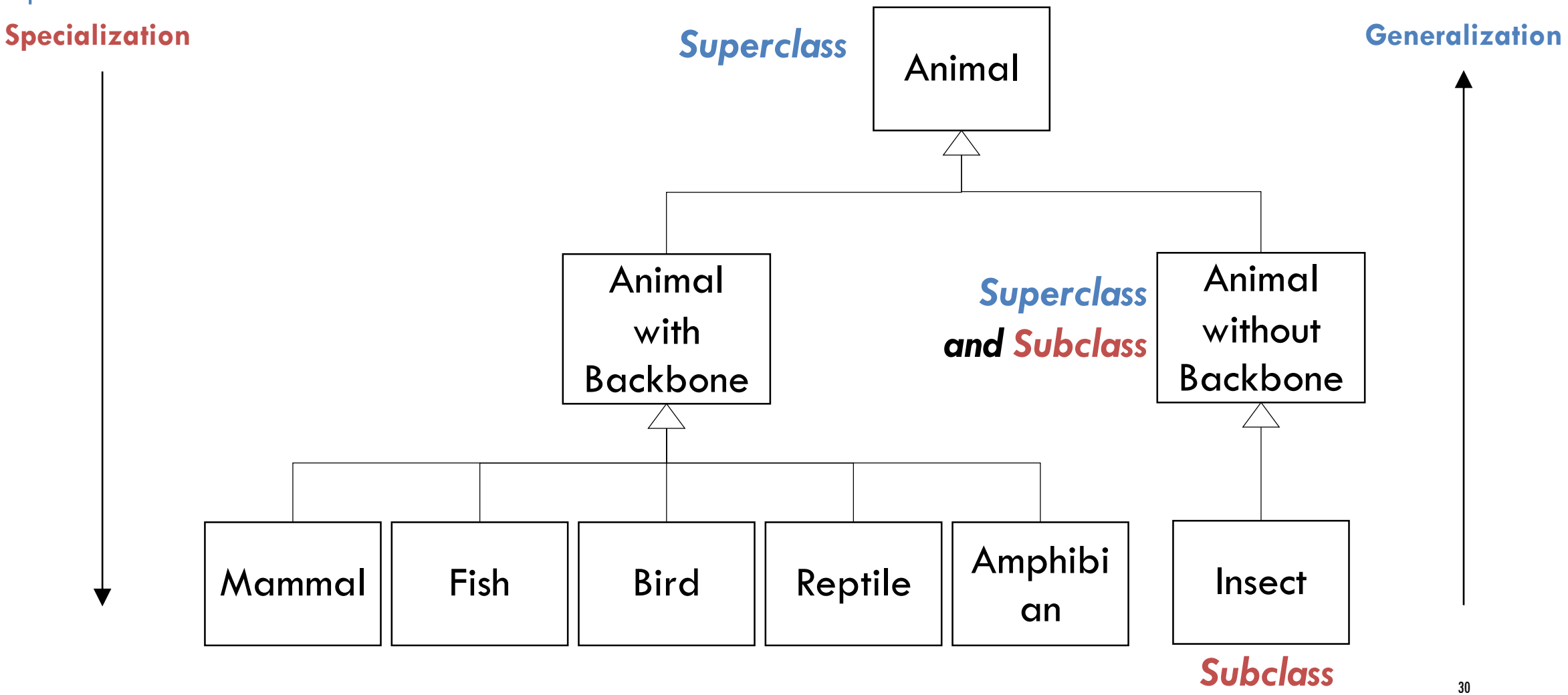
GENERALIZATION



SPECIALIZATION

- *specialization* เป็นกระบวนการตรวจจับความแตกต่างกันระหว่าง **object** ภายใน **class** จากนั้นสร้าง **subclass** ใหม่ที่ต่างจากเดิม
- เช่น **class Animal** ประกอบไปด้วย **object** มากมาย เมื่อเรามาตรวจสอบความแตกต่างทำให้พบว่า สัตว์เหล่านั้นแบ่งได้เป็น
  - Animal-with-Backbone
  - Animal-without-Backbone

GENERALIZATION AND SPECIALIZATION



ABSTRACT AND CONCRETE CLASSES

- **class** ที่มีคุณสมบัติทั่วไปที่จะไม่มีการนำไปสร้าง **object** เพื่อใช้งาน
- แต่จะสืบทอดคุณสมบัติ **attribute** และ **method** ไปยัง **subclass** เพื่อสร้าง **object** มาใช้งาน
- Class ที่มีลักษณะนี้เรียกว่า ***abstract class*** ในแนวคิดของ object-oriented modeling

ABSTRACT AND CONCRETE CLASSES

- เช่น **class Animal-with-Backbone** ได้รวม **class** อื่นๆที่มีคุณสมบัติที่คล้ายคลึงกันกับ **object** ของ **class Mammal, Fish, Bird, Reptile,** และ **Amphibian.**
- ในขณะเดียวกัน **class Animal-without-Backbone** ก็จะสื่อถึง **class insect**

ABSTRACT AND CONCRETE CLASSES

- Class ที่ทำหน้าที่อ้างถึงคุณสมบัติมักจะเป็น **class** ที่ไม่สร้าง **object** แต่ **class** ที่สร้าง **object** มาใช้งานมักจะเป็น **class** ที่อยู่ในระดับล่างของ Class Hierarchy Diagram
- Class ที่สามารถสร้าง **object** ได้เรียกว่า ***concrete classes***

ABSTRACT AND CONCRETE CLASSES

- สร้าง **class** ใหม่ดังนี้

```
public abstract class Animal {
    public void ShowAnimal(){
        System.out.println("I am Animal");
    }
}
```

```
public abstract class Animal_with_Backbone extends Animal {
    public void ShowAnimalWithBackbone(){
        System.out.println("I am Animal with Backbone");
    }
}
```

```
public abstract class Animal_without_Backbone extends Animal{
    public void ShowAnimalWithoutBackbone(){
        System.out.println("I am Animal without Backbone");
    }
}
```

ABSTRACT AND CONCRETE CLASSES

```
public class Mammal extends  Animal_with_Backbone {
    String name;
    public Mammal(String name){
        this.name = name;
    }
    public String getName(){
        return this.name;
    }
}
```

```
public class Reptile extends Animal_with_Backbone {
}
```

```
public class Amphibian extends Animal_with_Backbone {
}
```

```
public class Bird extends Animal_with_Backbone {
}
```

```
public class Insect extends Animal_without_Backbone {
}
```

ABSTRACT AND CONCRETE CLASSES

```
public class Main {

    public static void main(String[] args) {
        Bird b = new Bird();
        b.ShowAnimal();
        b.ShowAnimalWithBackbone();
    }
}
```

EXERCISES

1. ให้นักศึกษาแต่ละกลุ่มเลือกหัวข้อ keyword ต่อไปนี้

- กีฬา
- สัตว์เลี้ยง
- อาหาร
- ผลไม้
- ยานพาหนะ
- เครื่องใช้ไฟฟ้า
- เครื่องครัว
- เครื่องมือช่าง
- เครื่องสำอาง
- เครื่องแต่งกาย
- อุปกรณ์คอมพิวเตอร์

EXERCISES (ต่อ)

2. จากนั้นให้เขียน **class** ย่อยต่างๆที่สามารถสร้างได้จาก **keyword** ประมาณ 10-15 **class**

3. ทำการสร้าง **generalized class** จาก **class** ที่สร้างมา

4. สร้าง **class hierarchy diagram**

5. ระบุ *Concrete class* จาก **class hierarchy diagram**

สรุป