หน่วยที่ 4

การให้นิยามวัตถุและ UML Diagram (2)

Abstraction and UML Diagram (2)

เรื่องที่จะศึกษา

3.3 Generalization Abstraction

3.4 Association Abstraction

3.3 Generalization Abstraction

OOP Concept : Inheritance, Polymorphism

3.3 Generalization Abstraction

- อธิบายและใช้งาน Generalization Abstraction ได้
- อธิบายและใช้งาน Specialization Abstraction ได้
- อธิบายและใช้งาน Inheritance ได้

Generalization และ Specialization

- ในโลกแห่งความเป็นจริง วัตถุทุกชนิดจะแบ่งปันลักษณะหรือความเป็นอยู่ซึ่งกันและกัน
- เราสามารถจัดกลุ่มของวัตถุ ที่มีคุณสมบัติหรือลักษณะใกล้เคียงกัน ให้อยู่ในคลาสเดียวกัน
- ตัวอย่าง การจำแนก classes ของสัตว์...

ตัวอย่าง 7 คลาสของสัตว์

- Mammals
  - Dogs, cats, horses, duckbill platypuses, kangaroos, dolphins และ whales เป็นสัตว์เลี้ยงลูกด้วยนม
  - ตัวอ่อนดื่มนมเป็นอาหาร และมีขนตามร่างกาย
- Birds
  - นกมีขน และเกิดจากไขที่มีเปลือกแข็ง
  - ขนบนปีกและหาง จะทับซ้อนกันอยู่ ซึ่งทำให้โต้ลม และทำให้นกบินและร่อนลงได้

ตัวอย่าง 7 คลาสของสัตว์ (ต่อ)

- **Fish**
  - Fish เป็นสัตว์มีกระดูกสันหลัง
  - อาศัยในน้ำ มี เหงือก (gills), เกล็ด (scales) และ ครีบ (fins)
- **Reptiles (สัตว์เลื้อยคลาน)**
  - Reptiles เป็น class ของสัตว์ที่มีเกล็ดบนผิวหนัง
  - เป็นสัตว์เลือดเย็นและเกิดบนบก

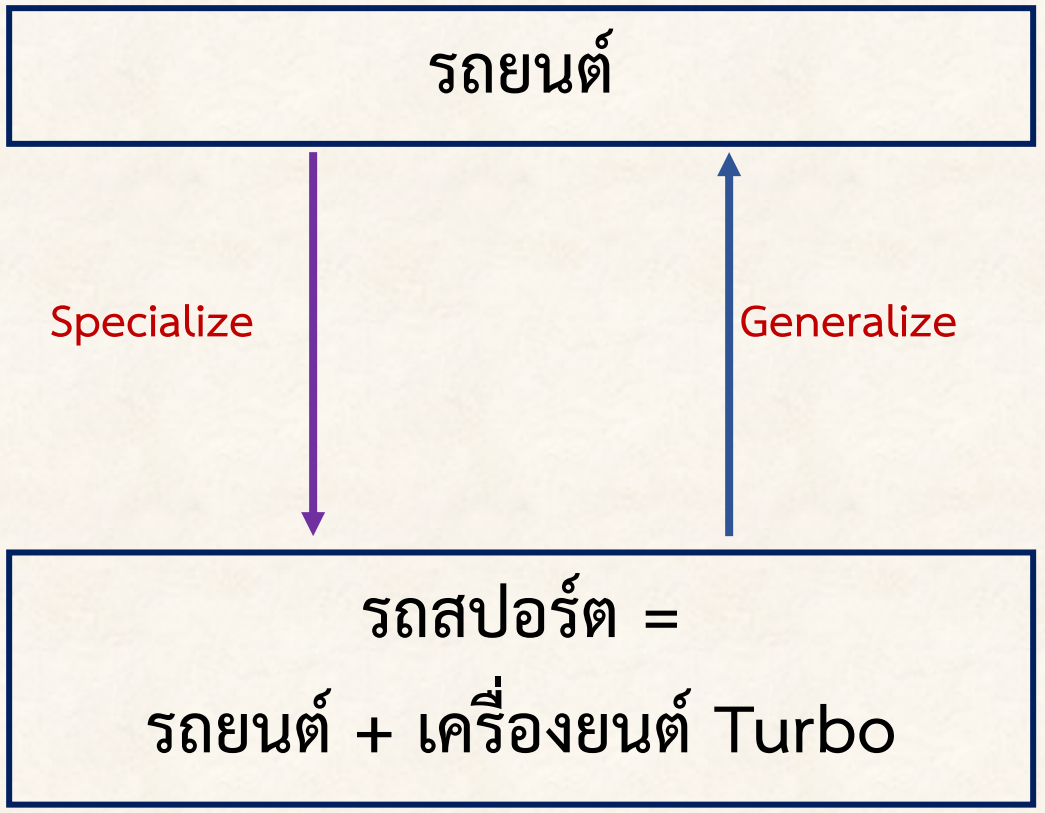
ตัวอย่าง 7 คลาสของสัตว์ (ต่อ)

- Amphibians (สัตว์ครึ่งบกครึ่งน้ำ)
  - Amphibians เกิดในน้ำ เมื่อแรกเกิดจะหายใจด้วยเหงือกคล้ายปลา
  - เมื่อโตขึ้นจะพัฒนาปอดขึ้นมาและอาศัยบนบกเป็นหลัก
- Arthropods
  - Arthropods เป็น class ของสัตว์ที่มีมากกว่า 4 ขา
  - แมลงต่างๆ รวมถึงแมงมุมอยู่ในประเภทนี้

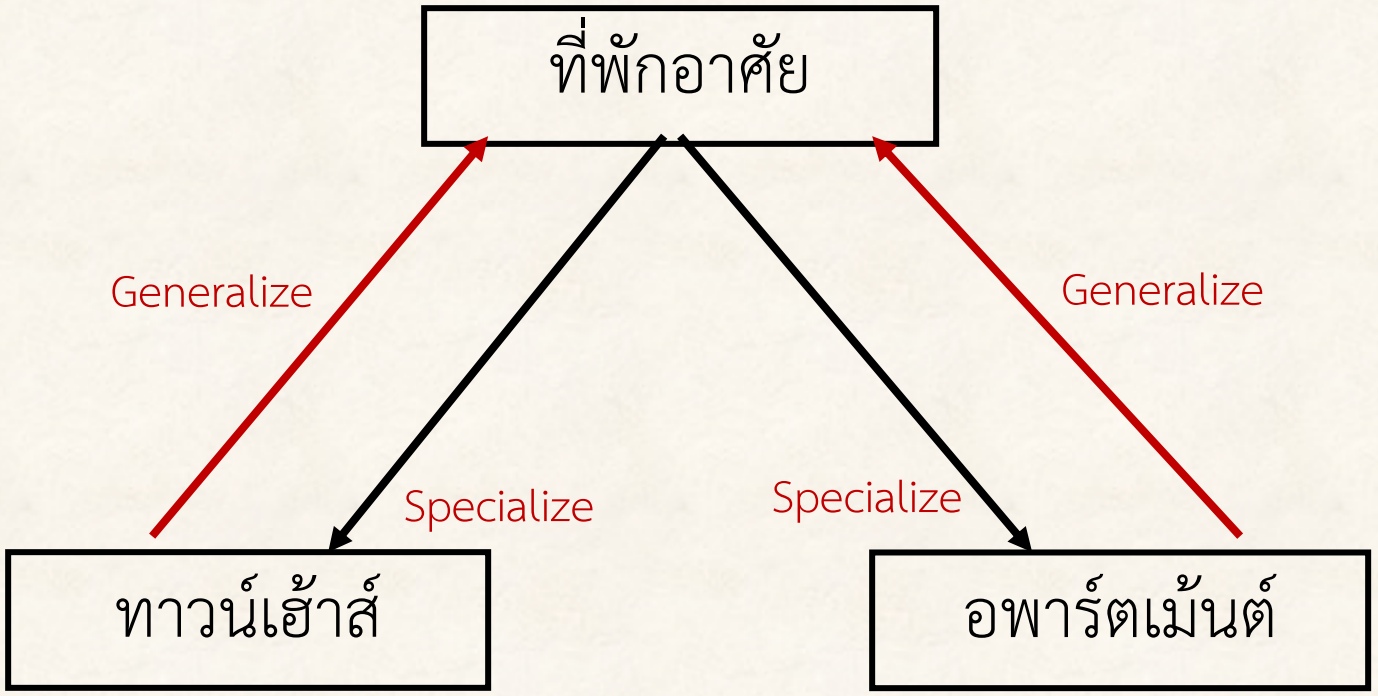
Generalization และ Specialization

- การให้ Concept ใหม่ กับ Class ใด Class หนึ่ง (จากหลายๆ Objects) โดยละเลยหรือตัดคุณสมบัติพิเศษบางอย่างออกไป เพื่อทำให้ Class ดังกล่าวมีลักษณะเป็นสามัญ (General) เรียกว่า กระบวนการ Generalize
- การให้ Concept ใหม่ กับ Class ใด Class หนึ่งที่มีอยู่แล้ว โดยพิจารณาหรือเพิ่มเติมคุณสมบัติใหม่ๆ ให้ Class (สามารถนำไปสร้างเป็น Object) ที่มีลักษณะพิเศษ (Special) เพิ่มขึ้นกว่าเดิม เรียกว่า กระบวนการ Specialize

ตัวอย่าง 8 รถยนต์และรถสปอร์ต



ตัวอย่าง 9 ที่พักอาศัย



กิจกรรม

- ให้นักศึกษาบอกถึงการทำ generalization และ specialization ที่พบในชีวิตประจำวันมาคนละกรณี

Inheritance

Inheritance

- กลไกที่เกิดจากการทำ Generalization Abstraction:
  - [Subclass-Superclass], [Base-Derived classes] และ Inheritance
  - กฎเกณฑ์ของการทำ Inheritance
  - Multiple Inheritance
  - Polymorphism

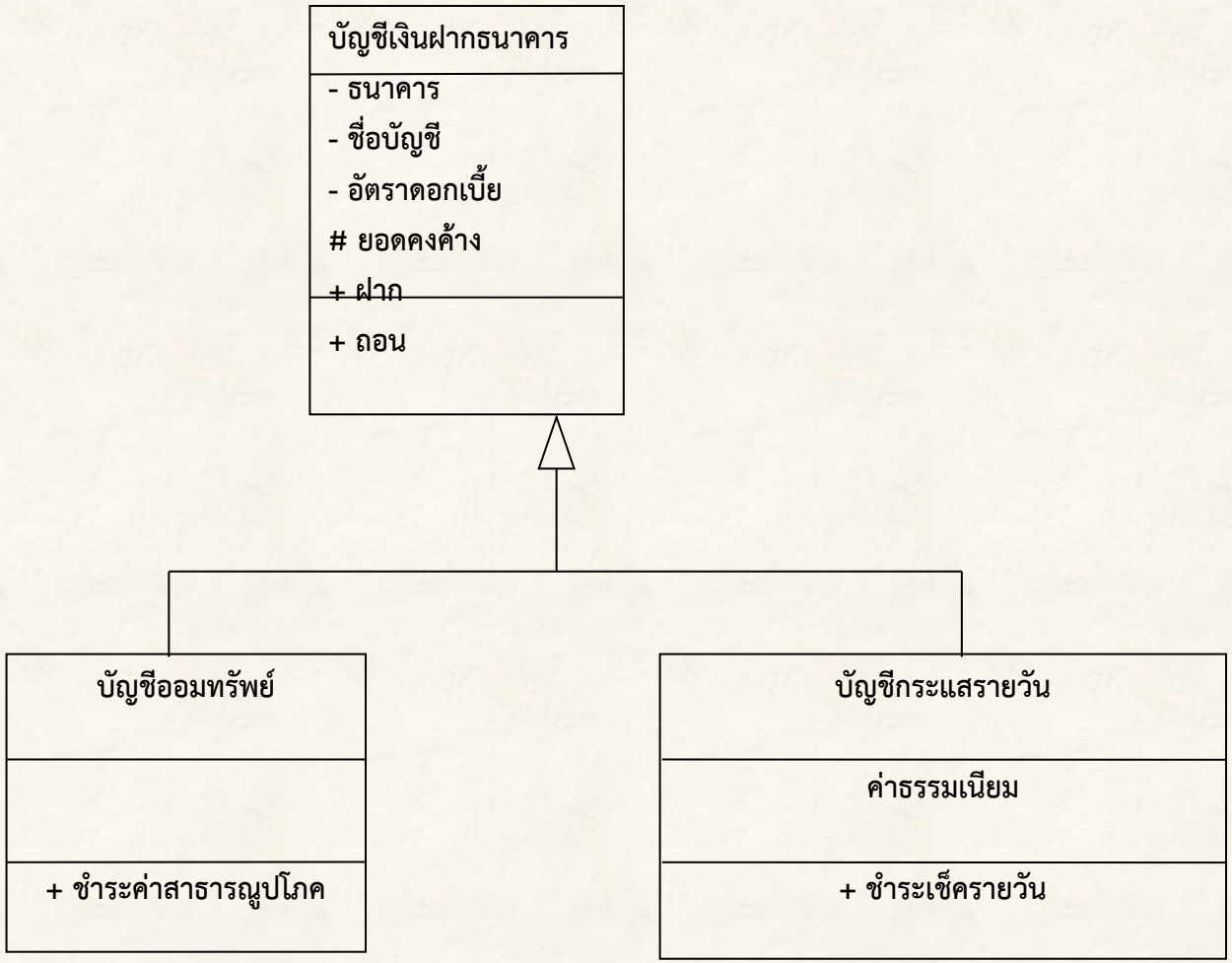
Subclass, Superclass และ Inheritance

- Superclass = Class เริ่มต้น หรือ class แม่ (base class)
- Subclass = Class ที่เกิดจากการทำ Specialize (derived class)
- Inheritance = กระบวนการ Specialization

ข้อควรจำในการทำ Inheritance

- Subclass ที่ inherit มาจาก superclass นั้นจะต้องมีคุณสมบัติทุกอย่างของ superclass (attribute & function) ผนวกกับคุณสมบัติพิเศษที่เพิ่มเข้าไปในแต่ละ subclass เสมอ
- ใช้สัญลักษณ์ลูกศรซึ่งมีหัวเป็นรูปสามเหลี่ยมใสชี้จาก subclass ไปยัง superclass

สัญลักษณ์แสดงการทำ Inheritance



กฎเกณฑ์ของการทำ Inheritance

- Private properties/methods จะถ่ายทอดมาเป็น Private properties/methods ของ derived class
  - ไม่สามารถเข้าถึง private attribute ที่ inherit มาได้
  - ไม่สามารถเข้าถึงได้จาก method ที่ไม่ได้มาจากการ Inheritance (ฟังก์ชันของ derived class เอง)

กฎเกณฑ์ของการทำ Inheritance (2)

- Protected properties/methods จะถ่ายทอดมาเป็น Protected properties/methods ของ derived class
  - สามารถเข้าถึงได้จากทุก methods ใน derived class

กฎเกณฑ์ของการทำ Inheritance (3)

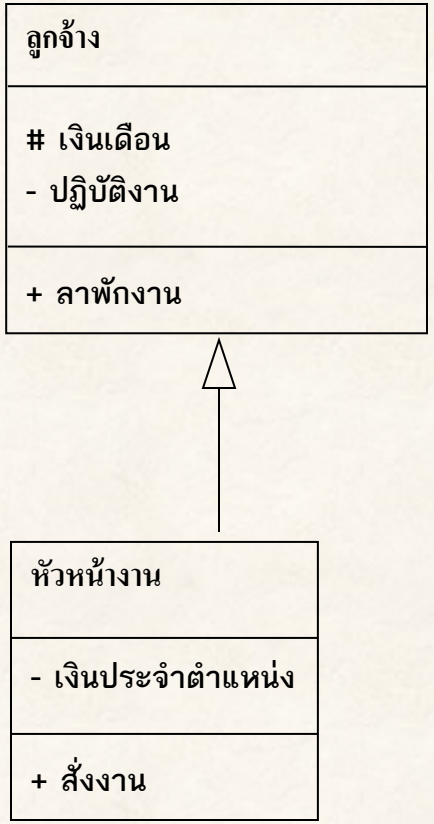
- Public properties/methods จะถ่ายทอดมาเป็น Public properties/methods ของ derived class เสมอ

ตัวอย่าง 10 Inside View ของ Class บัญชีออมทรัพย์ และ Class บัญชีกระแสรายวัน

บัญชีออมทรัพย์
- ธนาคาร - ชื่อบัญชี - อัตราดอกเบี้ย
ยอดคงค้าง + ฝาก + ถอน + ชำระค่าสาธารณูปโภค

บัญชีกระแสรายวัน
- ธนาคาร - ชื่อบัญชี - อัตราดอกเบี้ย
ยอดคงค้าง - ค่าธรรมเนียม + ฝาก + ถอน + ชำระเช็ครายวัน

ตัวอย่าง 11 Inheritance ของพนักงาน - หัวหน้างาน

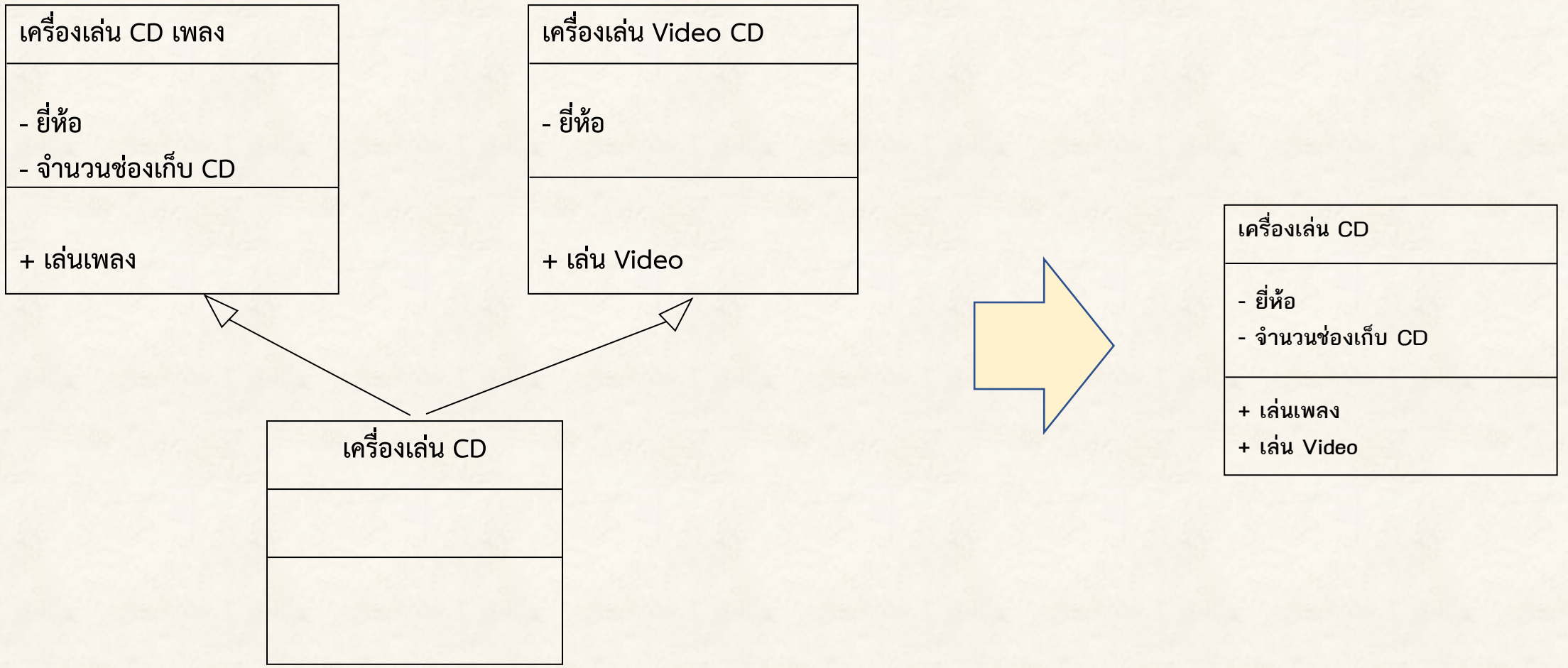


Multiple Inheritance

- เป็น Subclass ที่ inherit มาจาก superclass มากกว่า 1 ตัว เพื่อให้ได้ subclass ที่มีคุณสมบัติพิเศษเพียงตัวเดียว
- Subclass ที่เกิดจาก Multiple Inheritance นั้น จะเลือกเอา attribute และ function ที่ชื่อซ้ำกันจาก superclass ที่ได้ทำ inherit ก่อน เสมอ
- การทำ inherit จะทำ superclas ที่อยู่ ทางซ้าย ก่อนเสมอ

ภาษา C# ไม่รองรับการทำ multiple inheritance แต่มีกลไกอีกอย่างที่เรียกว่า Interface

ตัวอย่าง 12 Multiple Inheritance ของเครื่องเล่น CD

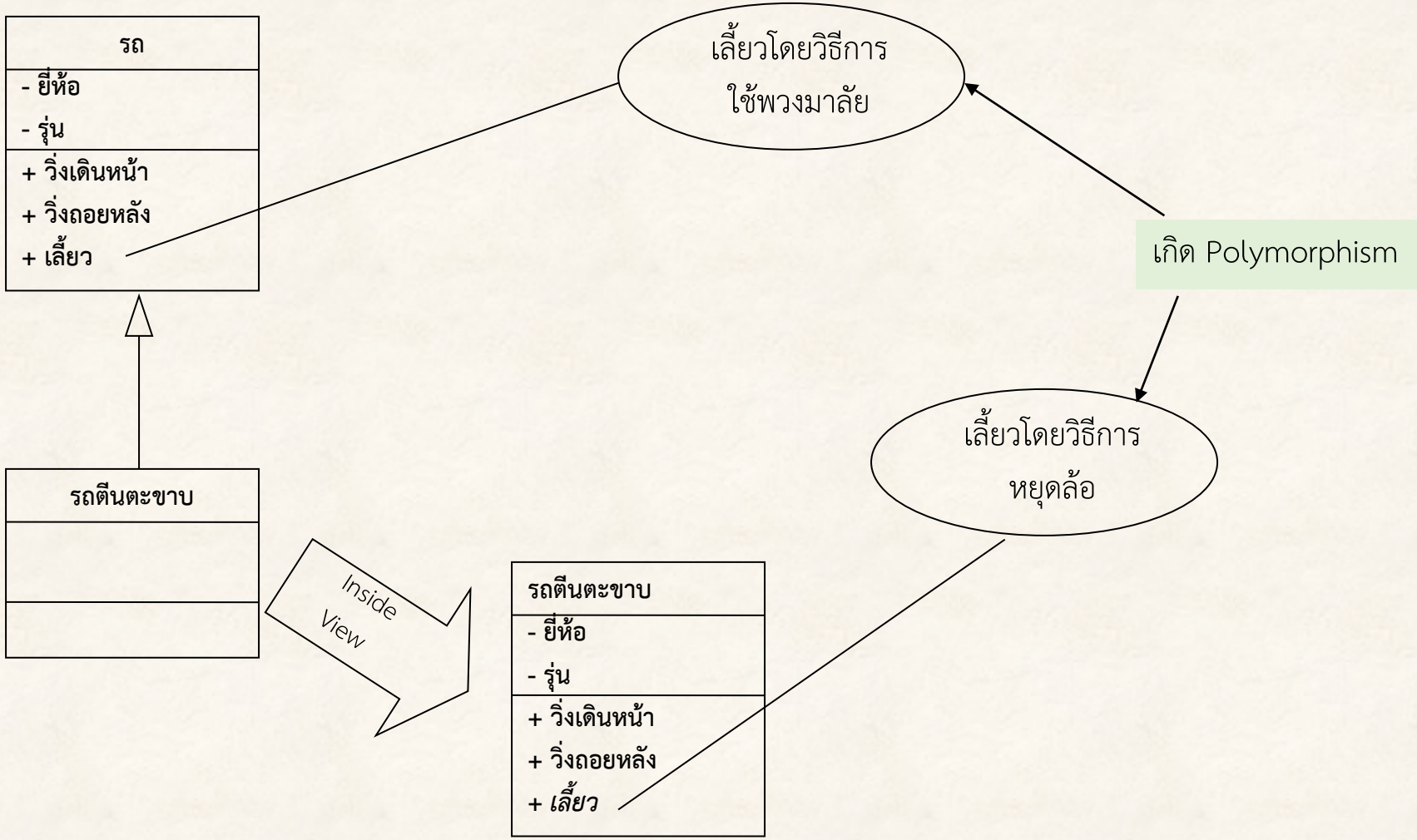


Polymorphism

Polymorphism

- เป็นคุณสมบัติของ Subclass ที่มีการดัดแปลง function บางอย่างโดยไม่ได้ยึดตาม superclass ทั้งหมด

ตัวอย่าง 13 ภาพจำลองแสดง Polymorphism



Association Abstraction

Association Abstraction

- เป็นการแสดงความสัมพันธ์ระหว่าง class ที่มีความสัมพันธ์แบบเกี่ยวพันกัน
- ไม่สามารถอธิบายโดย Abstraction แบบอื่นๆ ได้
  - ไม่ใช่ "Is a" แบบ Classification
  - ไม่ใช่ "Is part of" แบบ Aggregation
  - ไม่ใช่ "Is kind of" แบบ Generalization
- แต่เป็น "Is related to"

ตัวอย่าง 14 ความสัมพันธ์ ในโลกของความเป็นจริง

- คนเป็นเจ้าของรถยนต์
- แม่มีลูก
- สามีรักภรรยา
- ดินสออยู่ในกระเป๋า
- นักการเมืองออกกฎหมาย
- ทหารใส่เครื่องแบบ
- ประธานบริษัทบริหารกิจการ
- กระดานดำอยู่ในห้องเรียน

ตัวอย่าง 15 ความสัมพันธ์ ในโลกของความเป็นจริง

- พิจารณา “กระดานดำในห้องเรียน”
- เป็นได้ 2 แบบ คือ
  - Aggregation (กระดานดำเป็นองค์ประกอบของห้องเรียน) = เกิดมาพร้อมกัน
  - Association (กระดานดำเป็นครุภัณฑ์ประจำห้องเรียน) = นำมาประกอบภายหลัง

ตัวอย่าง 16 ความสัมพันธ์ ในโลกของความเป็นจริง

- “วันนี้ช่วงพักเที่ยง ให้นักศึกษาช่วยกันขนย้ายโต๊ะจากห้องเรียนไปไว้ที่ห้องทดลอง”
  - “โต๊ะเรียน” จะกลายสภาพเป็น “โต๊ะทดลอง”
  - เมื่อนำ “เครื่องมือทดลอง” มาวางบนโต๊ะทดลอง จะเป็นการเพิ่มความสามารถของโต๊ะ

Cardinality ใน Association Abstraction

- “Cardinality” คือตัวเลขที่ใช้แสดงจำนวนของสมาชิกที่สามารถมีได้ใน Class หนึ่งๆ ที่มีส่วนร่วมใน Association

ตัวอย่าง 17

- ผู้ชายมีภรรยาได้เพียง 1 คน หรือไม่มีเลยก็ได้ ในขณะเดียวกัน ผู้หญิงก็มีสามีได้เพียงคนเดียว หรือไม่มีเลยก็ได้

Class	Min Card	Max Card
ผู้หญิง	0	1
ผู้ชาย	0	1

ตัวอย่าง 18

- แม่สามารถมีลูกได้ตั้งแต่ 0 คน ถึงกี่คนก็ได้ ในทางกลับกัน ลูก 1 คน สามารถมีแม่ได้เพียงคนเดียว

Class	Min Card	Max Card
แม่	1	1
ลูก	0	N

ตัวอย่าง 19

- ใน 1 ภาคการศึกษา นักเรียนคนหนึ่งสามารถเรียนวิชาเรียนกี่วิชาก็ได้ (อย่างน้อยที่สุด 1 วิชา) ในขณะที่ วิชาหนึ่งๆ สามารถมีนักเรียนมาเรียนกี่คนก็ได้ (ในบางวิชาอาจไม่มีนักเรียนลงทะเบียนเรียนเลยก็ได้)

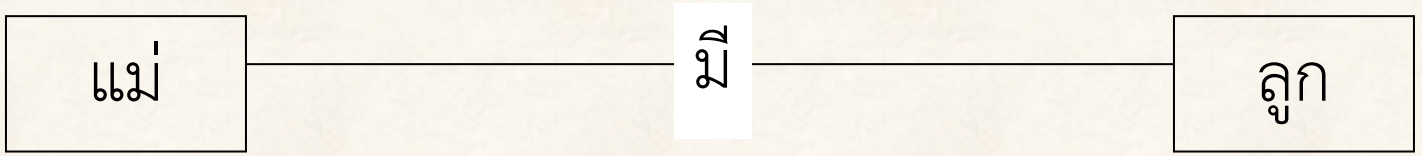
Class	Min Card	Max Card
นักเรียน	0	N
วิชาเรียน	1	N

หลักการในการเขียน Diagram แสดง Association

- เส้นตรงเชื่อมระหว่าง Class 2 Class
- มีลูกศรแสดงเส้นทางในการอ่านความสัมพันธ์
- มีชื่อของ Association กำกับที่เส้น
- มี Min Card และ Max Card ของ Class ทั้งสองกำกับที่ปลายเส้นด้านที่ติดกับ Class

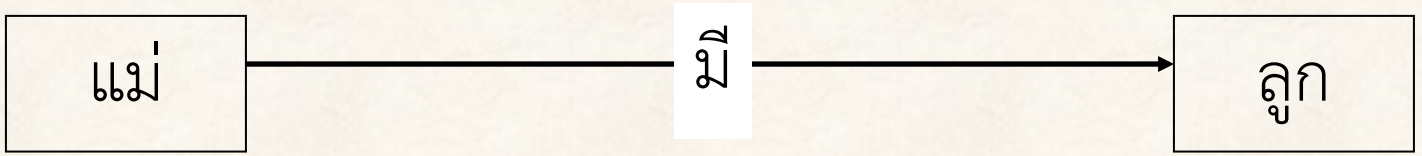
Association Abstraction ของ class แม่-ลูก

- ขั้นตอนที่ 1 : เขียน class 2 class ที่มีความสัมพันธ์และลากเส้นตรงใส่ชื่อแสดงความสัมพันธ์



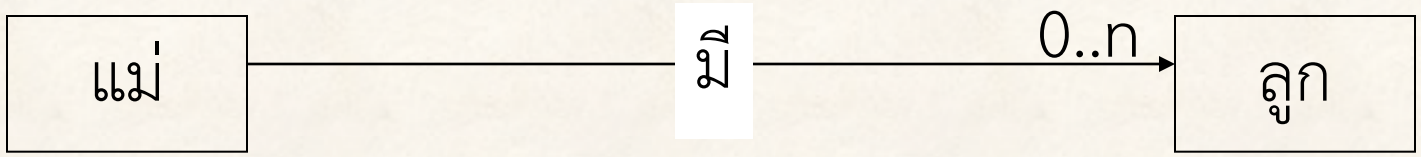
Association Abstraction ของ class แม่-ลูก

- ขั้นตอนที่ 2 : เขียนลูกศรเพื่อแสดงทิศทางของการอ่านความสัมพันธ์ให้ถูกต้อง



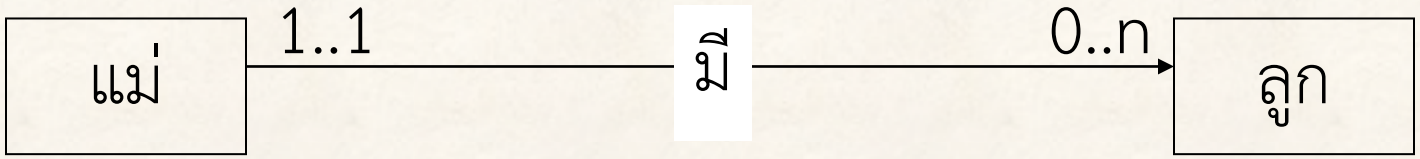
Association Abstraction ของ class แม่-ลูก

- ขั้นตอนที่ 3 : พิจารณา class ที่อยู่ติดกับหัวลูกศรว่ามีความสัมพันธ์กับ class แรกด้วย min-card และ max-card เป็นเท่าใด



Association Abstraction ของ class แม่-ลูก

- ขั้นตอนที่ 4 : พิจารณา class ที่อยู่ติดกับหัวลูกศรว่ามีความสัมพันธ์กับ class แรกด้วย min-card และ max-card เป็นเท่าใด จนกระทั่งได้ภาพที่สมบูรณ์



การบ้าน

References

- กิตติพงษ์ กลมกล่อม, "พื้นฐานการวิเคราะห์และออกแบบระบบเชิงวัตถุด้วย UML", สำนักพิมพ์ เคทีพี, 2552.
- พนิดา พานิชกุล, "การพัฒนาระบบเชิงวัตถุด้วย UML", สำนักพิมพ์ เคทีพี, 2552.
- พนิดา พานิชกุล, "Object-Oriented ฉบับพื้นฐาน" , สำนักพิมพ์ เคทีพี, 2548.