บทที่ 3
คลาสและการเขียนโปรแกรมเชิงวัตถุเบื้องต้น

3.1 ความสัมพันธ์ระหว่างคลาสและออบเจ็กต์

คลาส เป็นส่วนที่ใช้อธิบายโครงสร้างของออบเจ็กต์ ออบเจ็กต์หนึ่ง ๆ จะต้องถูกสร้างหรืออ้างอิงมาจากคลาส ดังนั้นคลาสจึงเป็นตัวกำหนดว่าออบเจ็กต์นั้นจะมี แอทริบิวต์ (Attribute) และ เมธอด (Method) อะไรบ้าง (Farrell, 2010 p.6) ซึ่งเป็นองค์ประกอบสำคัญของออบเจ็กต์

ในการเขียนโปรแกรมเชิงวัตถุนั้นจะไม่มีการใช้งานคลาสตรงๆ แต่จะใช้คลาสเป็นแม่แบบ (Template) เพื่อสร้างออบเจ็กต์ต่างๆ ขึ้นมา ออบเจ็กต์ที่สร้างมาจากคลาสบางครั้งจะเรียกว่าอินสแตนท์ (instance of) ของคลาสนั้น โดยคลาสหนึ่งสามารถใช้สร้างออบเจ็กต์ได้หลายออบเจ็กต์ แต่ออบเจ็กต์แต่ละตัวต้องมีชื่อที่แตกต่างกันไป (ธีรวัฒน์ ประกอบผล, 2554 หน้า 162) โดยขั้นตอนของการเกิดคลาส เริ่มต้นจากการจำแนกให้ได้ก่อนว่าสิ่งที่เรากำลังสนใจอยู่นั้นประกอบด้วยออบเจ็กต์อะไรบ้าง มีคุณลักษณะ (Attribute) อย่างไร และสามารถทำอะไรได้บ้าง (Method) ซึ่งเหล่านี้เรียกว่าเป็นการให้ Concept กับออบเจ็กต์ จากนั้นนำออบเจ็กต์ต่างๆ ที่ได้มาจัดกลุ่มโดยออบเจ็กต์ที่มีลักษณะเหมือนหรือคล้ายกันจะถูกจัดให้อยู่กลุ่มเดียวกัน หรืออีกทางก็คือจัดให้อยู่คลาสเดียวกัน

ตัวอย่างที่ 3.1 การวิเคราะห์ออบเจ็กต์และคลาส ภายในห้องเรียน (โดเมนตามภาพด้านล่าง)

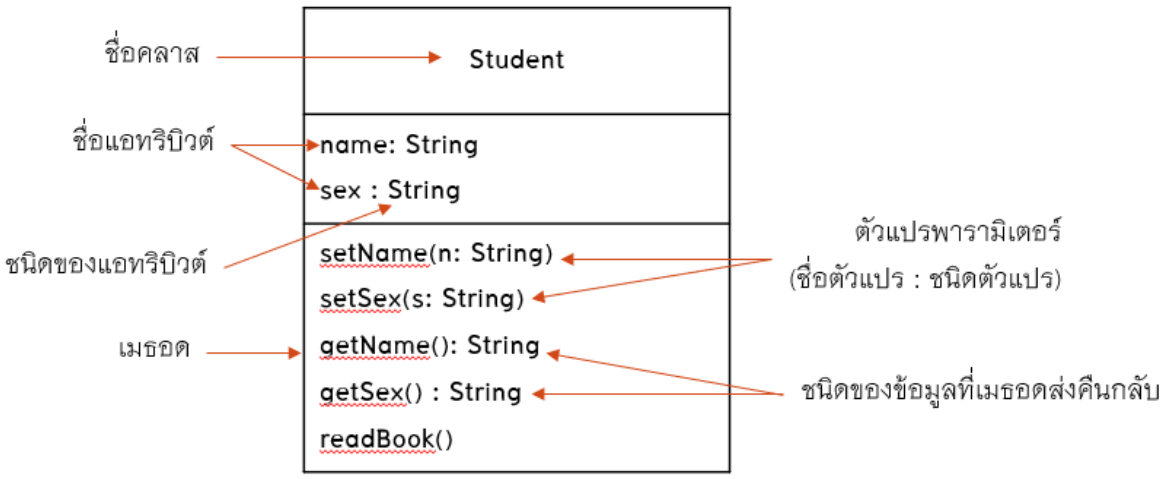
รูปที่ 3.1 ภาพตัวอย่างการวิเคราะห์คลาสและออบเจ็กต์

ขั้นตอนที่ 1 จำแนกออบเจ็กต์ : ออบเจ็กต์ มีทั้งหมด 33 ออบเจ็กต์ ประกอบด้วย ออบเจ็กต์ “นักเรียน” 2 ออบเจ็กต์ และ ออบเจ็กต์ “หนังสือ” 31 ออบเจ็กต์

ขั้นตอนที่ 2 ให้ Concept กับออบเจ็กต์ : ออบเจ็กต์นักเรียนทั้ง 2 ออบเจ็กต์กำหนดให้มีลักษณะ ที่เหมือนกัน คือ Attribute ประกอบด้วย ชื่อ–นามสกุล และเพศ ส่วน Method ประกอบด้วย ความสามารถในการบอกชื่อ–นามสกุล, บอกเพศ, และอ่านหนังสือ สำหรับออบเจ็กต์หนังสือ ทั้ง 31 ออบเจ็กต์กำหนดให้มีลักษณะที่เหมือนกัน คือ Attribute ประกอบด้วย หมายเลข ISBN, ชื่อหนังสือ, ชื่อผู้แต่ง และสีปก ส่วน Method ประกอบด้วย บอกหมายเลข ISBN, บอกชื่อหนังสือ, บอกชื่อผู้แต่ง, บอกสีปก, และห่อปก

ขั้นตอนที่ 3 จัดกลุ่มออบเจ็กต์: จากการให้ Concept กับแต่ละออบเจ็กต์ที่จำแนกมาได้นำมาจัดกลุ่มซึ่งพิจารณาจากลักษณะทั้ง คุณลักษณะ (Attribute) และ พฤติกรรม (Method) ของออบเจ็กต์ จะพบว่าจากทั้งหมด 33 ออบเจ็กต์ สามารถจัดกลุ่มออบเจ็กต์ได้เป็น 2 กลุ่ม ได้แก่ กลุ่มของออบเจ็กต์นักเรียน และกลุ่มออบเจ็กต์หนังสือ ซึ่งกลุ่มที่ได้จะนำไปสู่การสร้างคลาสสำหรับใช้สร้างออบเจ็กต์ ซึ่ง Attribute และ Method อาจจะมีมาเพิ่มเติมทีหลังได้ ตัวอย่างเช่น คลาสควรจะมีเมธอดสำหรับให้ออบเจ็กต์สามารถกำหนดคุณลักษณะเฉพาะของตัวเองที่แตกต่างจากออบเจ็กต์ที่สร้างจากคลาสเดียวกันได้ เช่น คลาสสำหรับสร้างออบเจ็กต์นักเรียนควรจะมีเมธอดสำหรับให้ออบเจ็กต์ใช้งานเพื่อกำหนดชื่อ–นามสกุล หรือเพศของตนเองได้

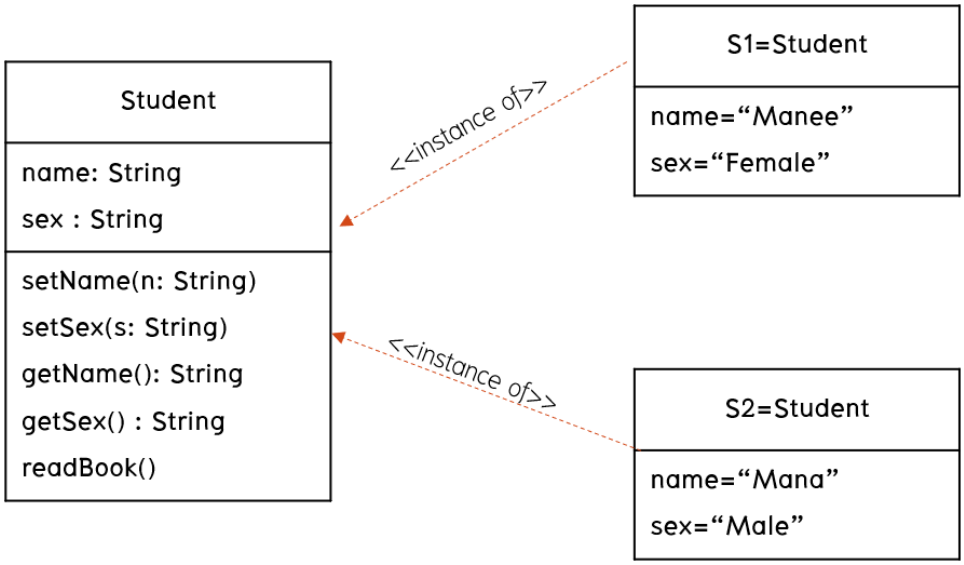
ในขั้นตอนของการวิเคราะห์ออกแบบโปรแกรมแนวคิดเชิงวัตถุ สามารถแสดงให้เห็นโครงสร้างของออบเจ็กต์ หรือคลาสได้โดยใช้แผนภาพชื่อว่า “คลาสไดอะแกรม (Class Diagram)” ดังรูปที่ 3.2



รูปที่ 3.2 ตัวอย่างคลาสไดอะแกรม (Class Diagram) ของคลาส Student

ที่มา : Oesterich, B., 2002

สำหรับแผนภาพที่ใช้แสดงถึงความสัมพันธ์ระหว่างออบเจ็กต์นักเรียนและคลาสนักเรียน สามารถเขียนแสดงได้ ดังตัวอย่างรูปที่ 3.3



รูปที่ 3.3 ตัวอย่างแผนภาพแสดงความสัมพันธ์ระหว่างคลาสและออบเจ็กต์

ที่มา: Oesterich, B., 2002

3.2 ชนิดของคลาส

คลาสในภาษาจาวา แบ่งได้เป็น 3 ชนิด (กิตติ ภักดีวัฒนะกุล, 2551) ได้แก่

1. Concrete Class เป็นคลาสทั่วไปซึ่งสามารถใช้งานเพื่อการสร้างออบเจ็กต์ได้ทันที

2. Abstract Class เป็นคลาสที่ถูกออกแบบมาให้ใช้งานในแบบลำดับชั้น แต่ไม่สามารถใช้งานเพื่อการสร้างออบเจ็กต์ได้ กล่าวคือ โครงสร้างของคลาสชนิดนี้จะมีแอทริบิวต์และเมธอด ซึ่งเมธอดที่มีสามารถมีได้ทั้งเมธอดที่มีระบุขั้นตอนการทำงาน (Method + Process) และ Abstract Method ซึ่งเป็นเมธอดที่มีระบุเฉพาะชื่อเท่านั้นแต่ไม่มีระบุขั้นตอนการทำงาน ทั้งนี้เพื่อให้คลาสอื่นที่เรียกใช้งาน Abstract Class นำไปเขียนขั้นตอนการดำเนินงานเอง ประโยชน์ของคลาสชนิดนี้ คือ ทำให้เราสามารถเขียนคลาสที่มีทั้งเมธอดที่มีขั้นการทำงาน และเมธอดที่ไม่มีขั้นตอนการทำงานแต่อนุญาตให้คลาสอื่น ๆ ที่เรียกใช้งานไประบุขั้นตอนการทำงานเองได้

3. Interface Class คลาสชนิดนี้คล้ายกันกับ Abstract Class แต่โครงสร้างภายในคลาสจะมีเฉพาะตัวแปรที่เป็นค่าคงที่ และเมธอดที่มีระบุเฉพาะชื่อเท่านั้น ทั้งนี้คลาสอื่นที่มาใช้งาน Interface Class จำเป็นต้องสร้างเมธอดที่มีชื่อตามที่ระบุไว้ใน Interface Class ด้วย

รายละเอียดของ Abstract Class และ Interface จะกล่าวถึงในบทถัดไป

3.3 Access Modifier และการใช้งาน

Access Modifier คือ ระดับในการเข้าถึง คลาส (Class), แอทริบิวต์ (Attribute) หรือเมธอด (Method) ในภาษาเชิงวัตถุ โดยกำหนดไว้หน้า Class, Attribute หรือ Method ซึ่งมีประโยชน์อย่างมากในเรื่องของการกำหนดระดับการเข้าถึง สิทธิ์ในการเข้าใช้งาน และการซ่อนข้อมูล ในภาษาจาวามี Access Modifier ทั้งหมด 4 ระดับ โดยมีการใช้งาน ดังตารางที่ 3.1

ตารางที่ 3.1 การใช้งาน **Access Modifier กับ**คลาส แอทริบิวต์ หรือเมธอด ในภาษาเชิงวัตถุ

Access Modifier	ใช้กับ Class	ใช้กับ Attribute	ใช้กับ Method
public	✓	✓	✓
protect		✓	✓
package	✓	✓	✓
private		✓	✓

ที่มา : อรพิน ประวัติบริสุทธิ์, (ม.ป.ป)

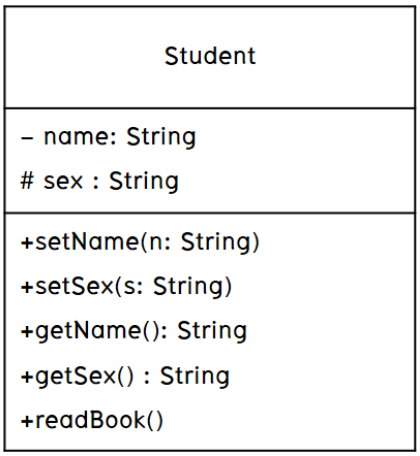
- public เป็นการประกาศระดับการเข้าถึงที่เข้มงวดน้อยที่สุด โดยคลาสอื่นๆ สามารถเรียกใช้งานคลาส แอตทริบิวต์ และเมธอดนั้นได้อย่างอิสระ
- protected เป็นการประกาศระดับการเข้าถึงที่เกี่ยวข้องกับเรื่องการสืบทอด (Inheritance) คลาสที่อยู่ในแพ็คเกจเดียวกัน กับคลาสที่มีสมาชิกที่ถูกกำหนดเป็น protected เท่านั้นที่จะสามารถเรียกใช้งานสมาชิกตัวนั้นได้ ในกรณีที่คลาสมีการสืบทอดคุณสมบัติจากคลาสที่สมาชิกถูกกำหนดเป็นแบบ protected สามารถเรียกใช้งานสมาชิกตัวนั้นได้ ถึงแม้จะอยู่ต่างแพ็คเกจกัน
- package นับเป็นระดับการเข้าถึงเริ่มต้น (Default) ในภาษาจาวา กรณีที่ Class, Attribute หรือ Method ไม่ได้กำหนด Access Modifier ใดๆ ไว้ข้างหน้า โดยคลาสที่อยู่ต่างแพ็คเกจกันจะไม่สามารถเรียกใช้งานสมาชิกของกันและกันได้
- private เป็นการประกาศระดับการเข้าถึงที่เข้มงวดมากที่สุด โดยคลาสอื่นๆ จะไม่สามารถเรียกใช้งาน แอตทริบิวต์หรือเมธอด เฉพาะภายในคลาส ที่แอตทริบิวต์และเมธอดนั้นอยู่เท่านั้น คลาสอื่นๆ ไม่สามารถเรียกใช้งานได้

ตารางที่ 3.2 การทำงานของ Access Modifier แต่ละแบบ

Access Modifier	ใช้ได้ทั้งหมด	package เดียวกัน	ต่าง package กัน	ต่าง package กัน แต่เป็นคลาสลูกกัน	Class เดียวกัน
public	✓	✓	✓	✓	✓
protect	X	✓	✓	✓	✓
package	X	✓	X	X	✓
private	X	X	X	X	✓

ที่มา : อรพิน ประวัติบริสุทธิ์, (ม.ป.ป)

ในแผนภาพ Class Diagram สามารถแสดงสัญลักษณ์ Access Modifier ที่ใช้กับ Attribute และ Method ของคลาส ได้โดย Private ใช้เครื่องหมายลบ (–), Protected ใช้เครื่องหมาย Shaft (#), และ Public ใช้เครื่องหมายลบ (+) ส่วน Package นับเป็นระดับการเข้าถึงเริ่มต้นไม่จำเป็นตอ้งใช้สัญลักษณ์ใดในการระบุ ตัวอย่างการใช้งานสัญลักษณ์ Access Modifier ในแผนภาพ Class Diagram ดังรูปที่ 3.4



รูปที่ 3.4 ตัวอย่างการใช้งานสัญลักษณ์ Access Modifier ในแผนภาพ Class Diagram

3.4 การสร้างคลาส

3.4.1 การประกาศคลาส

การสร้างคลาสหรือการประกาศคลาสในภาษาจาวา (ธีรวัฒน์ ประกอบผล, 2554) มีรูปแบบดังนี้

```
[AccessModifier] class Name
{
        Members
}
```

- AccessModifier ระดับการเข้าถึงคลาส (public, package)
- class เป็นคีย์เวิร์ด (keyword) ใช้ในการประกาศคลาส
- Name เป็นชื่อคลาสที่ประกาศขึ้น
- Members เป็นแอทริบิวต์และเมธอดต่าง ๆ ซึ่งเป็นองค์ประกอบของออบเจ็กต์

กฎและข้อแนะนำในการตั้งชื่อ Class

- กฎการตั้งชื่อคลาสเป็นไปตามกฎการตั้งชื่อ identify ของภาษาจาวา
- นิยมขึ้นต้นด้วยตัวใหญ่
- หากชื่อคลาสประกอบด้วยคำมากกว่า 1 คำ โดยทั่วไปนิยมกำหนดให้ตัวอักษรแรกของแต่ละคำเป็นตัวใหญ่
- ชื่อคลาสควรเป็นคำนาม

ตัวอย่างเช่น

ตัวอย่างชื่อคลาสที่ถูกต้อง	
✓	Employee
✓	UnderGradStudent
✓	InventoryItem
✓	Budget2011
✓	Inventory_Item

ตัวอย่างชื่อคลาสที่ผิดกฎ	
X	An employee
X	Inventory Item
X	class
X	2011Budget
X	phone#

รูปที่ 3.5 ตัวอย่างการตั้งชื่อคลาสที่ถูกต้อง

3.4.2 การประกาศแอทริบิวต์

แอทริบิวต์ (Attribute) เป็นสมาชิกที่เป็นข้อมูลของคลาส ใช้ในการอธิบายถึงคุณลักษณะของออบเจ็กต์การประกาศแอทริบิวต์ ก็คือ การประกาศตัวแปรในภาษาจาวา (ธีรวัฒน์ ประกอบผล, 2554) มีรูปแบบ ดังนี้

```
[AccessModifier] dataType Name;
```

- Access*Modifier* เป็นความสามารถในการเข้าถึงแอทริบิวต์ ซึ่งจะใช้ public หากต้องการให้ออบเจ็กต์อื่นสามารถใช้งานได้ หรือจะใช้ private หากไม่ต้องการให้ออบเจ็กต์เข้าถึงได้โดยตรง
- dataType เป็นประเภทของแอทริบิวต์ ซึ่งก็คือประเภทของข้อมูลโดยทั่วไปในภาษาจาวา
- Name เป็นชื่อแอทริบิวต์ที่ประกาศขึ้น

หากต้องการกำหนดค่าเริ่มต้นให้กับแอทริบิวต์ สามารถกำหนดได้โดยมีรูปแบบ ดังนี้

```
attribute_Name = ค่าที่ต้องการกำหนดให้แอทริบิวต์;
```

กฎและข้อแนะนำในการตั้งชื่อ Attributes

- กฎการตั้งชื่อแอตทริบิวต์เป็นไปตาม กฎการตั้งชื่อ identify ของภาษาจาวา
- นิยมขึ้นต้นด้วยตัวเล็ก
- หากชื่อแอตทริบิวต์ประกอบด้วยคำมากกว่า 1 คำ โดยทั่วไปนิยมกำหนดให้ตัวอักษรตัวแรกของแต่ละคำ (ยกเว้นคำแรก) เป็นตัวใหญ่
- ชื่อแอตทริบิวต์ควรเป็นคำนาม

ตัวอย่างเช่น

ตัวอย่างชื่อแอตทริบิวต์ที่ถูกต้อง		ตัวอย่างชื่อแอตทริบิวต์ที่ผิดกฎ	
✓	employeeId	X	position^_^
✓	student_Name	X	inventory Item
✓	salary$	X	void
✓	budget2011	X	2011Budget
✓	Inventory_Item	X	phone#

รูปที่ 3.6 ตัวอย่างการตั้งชื่อแอทริบิวต์ที่ถูกต้อง

3.4.3 การประกาศเมธอด

เมธอด (Method) เป็นสมาชิกที่เป็นความสามารถของคลาส ใช้ในการอธิบายถึงพฤติกรรมของออบเจ็กต์การประกาศเมธอดในภาษาจาวา มีรูปแบบ ดังนี้

```
[AccessModifier] returnType Name(arguments)
{
        statement;
        return (data);
}
```

- Access*Modifier* เป็นความสามารถในการเข้าถึงเมธอด ซึ่งจะใช้ public หากต้องการให้ออบเจ็กต์อื่นสามารถใช้งานได้ หรือจะใช้ private หากไม่ต้องการให้ออบเจ็กต์อื่นเข้าถึงได้โดยตรง หรือจะใช้ protected หากไม่ต้องการให้ออบเจ็กต์อื่นนอกจากออบเจ็กต์ที่เป็น subclass เข้าถึงได้โดยตรง
- returnType เป็นชนิดของข้อมูลที่เมธอดจะส่งคืนค่ากลับ ซึ่งก็คือประเภทของข้อมูลในภาษาจาวาทั่วไป หากเมธอดไม่มีการส่งคืนค่ากลับ ให้ระบุคีย์เวิร์ด void ลงไป
- Name เป็นชื่อเมธอดที่ประกาศขึ้น
- arguments เป็นตัวแปรอาร์กิวเมนต์ หรือเรียกอีกอย่างหนึ่งว่าตัวแปรพารามิเตอร์ ใช้ในการรับข้อมูลเข้าเมธอด ซึ่งจะมีหรือไม่มีก็ได้
- statement เป็นคำสั่งต่าง ๆ ที่เขียนขึ้นภายในเมธอด
- return เป็นคีย์เวิร์ดที่ใช้สำหรับการส่งคืนค่ากลับของเมธอด ซึ่งจะเป็นคำสั่งที่อยู่บรรทัดสุดท้ายเสมอ ถ้า returnType เป็น void ก็ไม่ต้องมีการระบุส่วน return นี้
- (data) คือข้อมูลที่เมธอดจะส่งคืนค่ากลับจะอยู่ในรูปของข้อมูลหรือตัวแปรก็ได้

กฎและข้อแนะนำในการตั้งชื่อ Methods

- กฎการตั้งชื่อเมธอดเป็นไปตาม กฎการตั้งชื่อ identifier ของภาษาจาวา
- นิยมขึ้นต้นด้วยตัวเล็ก
- หากชื่อเมธอดประกอบด้วยคำมากกว่า 1 คำ โดยทั่วไปนิยมกำหนดให้ตัวอักษรตัวแรกของแต่ละคำ (ยกเว้นคำแรก) เป็นตัวใหญ่
- ชื่อเมธอดควรกำหนดให้เป็นคำกริยา

ตัวอย่างเช่น

ตัวอย่างชื่อเมธอดที่นิยมและถูกต้อง	
✓	getName()
✓	openMobile()
✓	closeMobile()
✓	setRingTone()
✓	getPhone()

ตัวอย่างชื่อเมธอดที่ผิดกฎ	
X	getName ()
X	get Time()
X	static()
X	2_()
X	getPhone#()

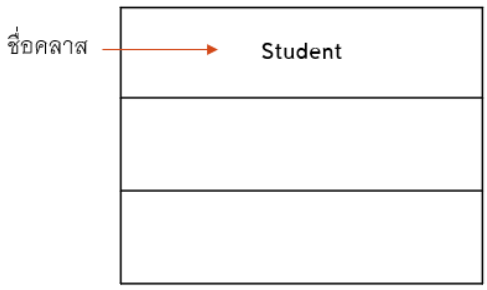
รูปที่ 3.7 ตัวอย่างการตั้งชื่อเมธอดที่ถูกต้อง

3.4.4 ตัวอย่างการสร้างคลาส

1. การประกาศคลาส

```
public class Student {

}
```



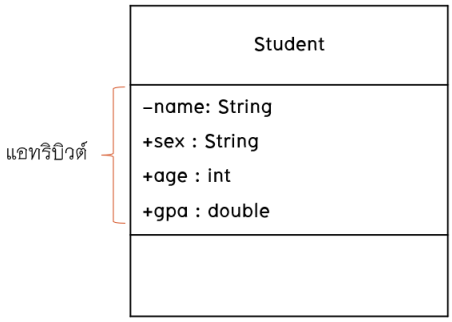
ตัวอย่างเป็นเพียงการประกาศคลาส Student ขึ้นมา เท่านั้น โดยกำหนด Access modifier เป็น public ซึ่งหมายถึงคลาสอื่นๆ สามารถมองเห็นและเรียกใช้งานคลาส Student นี้ได้

รูปที่ 3.8 คลาสไดอะแกรมของคลาส Student

2. การประกาศแอทริบิวต์

การประกาศแอทริบิวต์ต้องประกาศไว้ภายในคลาส แต่อยู่นอกเมธอด ดังนี้

```
public class Student {
        private String name;
        public String sex;
        public int age;
        public double gpa;
}
```



รูปที่ 3.9 คลาสไดอะแกรมของคลาส Student ที่มีแอทริบิวต์

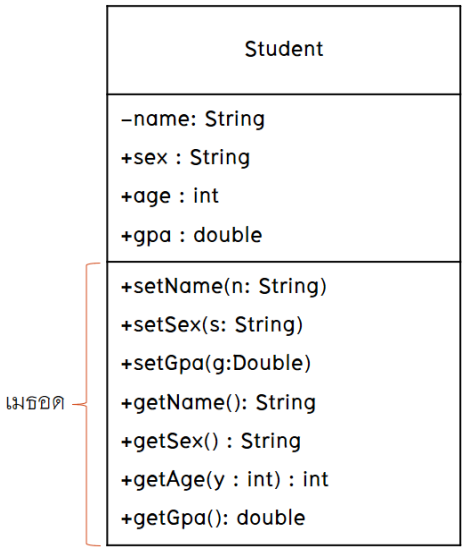
ตัวอย่างเป็นการประกาศคลาส Student มีแอทริบิวต์ ชื่อ (name) ที่มี Access Modifier เป็น private และมีแอทริบิวต์เพศ (sex), อายุ (age) และเกรดเฉลี่ยสะสม (gpa) ที่มี Access Modifier เป็น public ด้วย

3. การประกาศเมธอด

การประกาศเมธอดเหมือนกับการประกาศแอทริบิวต์ คือ ต้องประกาศไว้ภายในคลาส จากคลาส Student ก่อนหน้าที่ได้มีการประกาศแอทริบิวตีไปแล้ว เราสามารถประกาศเมธอดเพิ่มเข้าไปได้ดังนี้

```
public class Student [
        private String name, sex;
        public int age;
        public double gpa;

        public void setName(String n) {
                name=n;
        }
        public void setSex(String s) {
                sex=s;
        }
        public void setGpa (double g) {
                gpa=g;
        }
        public String getName() {
                return name;
        }
        public String getSex(){
                return sex;
        }
        public int getAge(int y){
                age = 2560-y;
                return age;
        }
        public double getGpa(){
                return gpa;
        }
```



รูปที่ 3.10 คลาสไดอะแกรมของคลาส Student ที่มีแอทริบิวต์และเมธอด

ตัวอย่างเป็นการประกาศเมธอดเพิ่มขึ้นมาในคลาส Student รายละเอียด ดังนี้ เมธอด setname(), setSex(), setGpa() เป็นเมธอดที่ไม่มีการส่งคืนค่ากลับ แต่มีการรับค่าอาร์กิวเม้นต์เข้าเมธอดจำนวน 1 ตัวตอนออบเจ็กต์เรียกใช้งานเมธอดนั้น เพื่อนำค่ามาจัดเก็บในแอทริบิวต์ ซึ่งเมธอด setName() จะจัดเก็บข้อมูลใน name เมธอด setSex() จะจัดเก็บข้อมูลใน sex และเมธอด setGpa() จะจัดเก็บข้อมูลใน gpa สำหรับเมธอด getName(), getSex(), getAge(), และ getGpa() เป็นเมธอดที่มีการส่งคืนค่าของแอทริบิวต์ กลับให้แก่ออบเจ็กต์ที่เรียกใช้งานเมธอด โดยเมธอด getName() จะส่งคืนค่าแอทริบิวต์ name เมธอด getSex() จะส่งคืนค่าแอทริบิวต์ sex เมธอด getAge() จะรับค่าอาร์กิวเม้นต์จากออบเจ็กต์ เพื่อมาคำนวณหาค่าอายุ และส่งคืนค่าที่คำนวณได้ให้กับออบเจ็กต์ และเมธอด getGpa() จะส่งคืนค่าแอทริบิวต์ gpa

3.5 การสร้างออบเจ็กต์

หลังจากนิยามคลาสขึ้นมาแล้ว ถ้าหากโปรแกรมต้องการใช้งานจะต้องสร้างออบเจ็กต์ขึ้นมาเพื่อบอกว่าออบเจ็กต์เป็นของคลาสใด รูปแบบ ดังนี้

```
ClassName ObjectName;
```

- ClassName คือชื่อคลาสที่ต้องการเรียกใช้งาน
- Objectname คือชื่อตัวแปรออบเจ็กต์

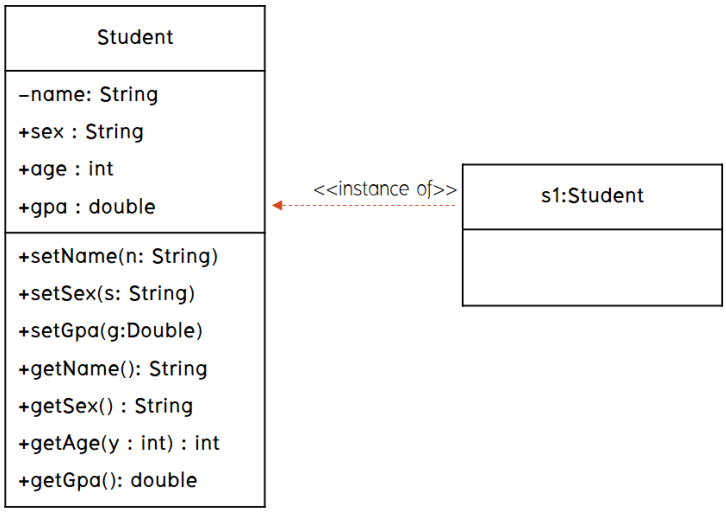
จากนั้นให้เขียนคำสั่งสร้างออบเจ็กต์ขึ้นมา โดยใช้คีย์เวิร์ด new รูปแบบดังนี้

```
ObjectName = new ClassName(argruments);
```

- Objectname คือชื่อตัวแปรออบเจ็กต์
- ClassName คือชื่อคลาสที่ต้องการเรียกใช้งาน
- argruments คือตัวแปรอาร์กิวเม้นต์ที่คลาสใช้เพื่อรับข้อมูลเข้าไปใช้ในคลาส (ซึ่งคลาสอาจจะมีหรือไม่มีก็ได้)

ตัวอย่างเช่น ตามแผนภาพรูปที่ 3.11 สามารถเขียนคำสั่งสร้างออบเจ็กต์ของคลาส Student ได้ดังนี้

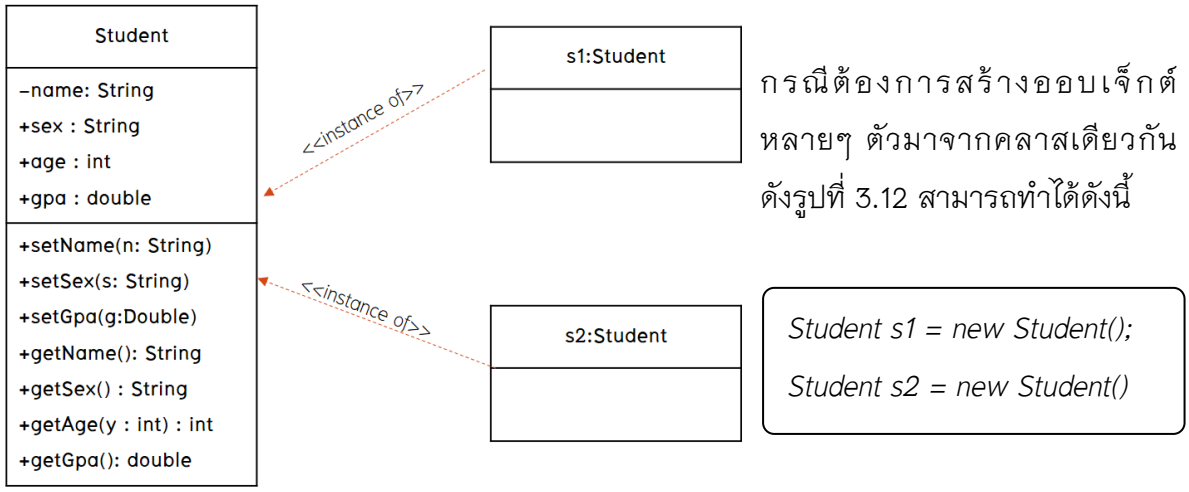
```
Student s1;
s1 = new Student();
```



รูปที่ 3.11 แผนภาพแสดงความสัมพันธ์ระหว่างออบเจ็กต์ s1 กับคลาส Student

หรืออีกวิธีเราสามารถรวมคำสั่งการประกาศและการสร้างออบเจ็กต์เข้าไว้ด้วยกันได้ ตัวอย่างเช่น

```
Student s1 = new Student;
```



กรณีต้องการสร้างออบเจ็กต์หลายๆ ตัวมาจากคลาสเดียวกัน ดังรูปที่ 3.12 สามารถทำได้ดังนี้

รูปที่ 3.12 แผนภาพแสดงความสัมพันธ์ระหว่างออบเจ็กต์ s1 และ s2 กับคลาส Student

3.6 การใช้งานแอทริบิวต์และเมธอด

แอทริบิวต์และเมธอดเป็นองค์ประกอบที่อยู่ภายในคลาส การใช้งานจึงต้องกระทำผ่านตัวแปรออบเจ็กต์ที่สร้างขึ้นมาจากคลาสนั่นเอง โดยที่ออบเจ็กต์จะสามารถเรียกใช้งานแอทริบิวต์และเมธอดที่สามารถมองเห็นได้เท่านั้น หรือกล่าวอีกอย่าง คือ ออบเจ็กต์จะสามารถเรียกใช้งานแอทริบิวต์และ เมธอด ตัวที่ Access Modifier ไม่ได้ถูกกำหนดให้เป็น private ของคลาสได้เท่านั้น ดังนี้

3.6.1 การเรียกใช้งานแอทริบิวต์

รูปแบบคำสั่งของการเรียกใช้งานแอทริบิวต์ของออบเจ็กต์ สามารถเขียนได้ ดังนี้

```
objectName.attributeName;
```

ออบเจ็กต์จะการเรียกใช้งานแอทริบิวต์ภายในคลาสได้ด้วยการเขียนคำสั่งนำหน้าด้วยชื่อออบเจ็กต์ แล้วคั่นด้วยเครื่องหมายจุด (.) และตามด้วยชื่อแอทริบิวต์ที่ต้องการใช้งาน ตัวอย่างเช่น

```
s1.sex = “Femal” ;
s2.gpa = 3.55 ;
```

จากตัวอย่างเป็นการบอกให้ออบเจ็กต์ที่สร้างขึ้นจากคลาส Student เรียกใช้งานแอทริบิวต์ โดยออบเจ็กต์ s1 เรียกใช้งานแอทริบิวต์ sex เพื่อกำหนดค่าให้เท่ากับ “Femal” และให้ออบเจ็กต์ s2 เรียกใช้งานแอทริบิวต์ gpa เพื่อกำหนดค่าให้เท่ากับ 3.55

3.6.2 การเรียกใช้งานเมธอด

รูปแบบคำสั่งของการเรียกใช้งานเมธอดของออบเจ็กต์จะเหมือนเดียวกันกับการเรียกใช้งานแอทริบิวต์ คือ เขียนคำสั่งนำหน้าด้วยชื่อออบเจ็กต์ แล้วคั่นด้วยเครื่องหมายจุด (.) และตามด้วยชื่อเมธอดที่ต้องการใช้งาน ดังนี้

```
objectName.methodName();
```

ตัวอย่างเช่น

```
s1.setName(“Sudarat Arthan”);
System.out.println (s1.getName());
System.out.println (s2.getAge(2533));
```

ผลลัพธ์แสดงออกทางจอภาพ

```
Sudarat Arthan
27
```

จากตัวอย่างเป็นการบอกให้ออบเจ็กต์ที่สร้างขึ้นจากคลาส Student เรียกใช้งานเมธอด ภายในคลาส โดยออบเจ็กต์ s1 เรียกใช้งานเมธอด setName เพื่อกำหนดชื่อเท่ากับ “Sudarat Arthan” และเรียกใช้เมธอด getName() เพื่ออ่านชื่อที่กำหนดโดยออบเจ็กต์ s1 แล้วพิมพ์แสดงผลออกทางจอภาพ และให้ออบเจ็กต์ s2 เรียกใช้งานเมธอด getAge() เพื่ออ่านค่าอายุของออบเจ็กต์ แล้วพิมพ์แสดงผลออกทางจอภาพ โดยตอนเรียกใช้งานเมธอดออบเจ็กต์ s2 จะมีการส่งค่า 2533 ซึ่งเป็นค่าอาร์กิวเมนต์ที่เมธอดต้องนำไปใช้ประมวลผลไปด้วย

3.7 บทสรุป

ในบทที่ 3 เป็นการกล่าวถึงความสัมพันธ์ระหว่างคลาสและออบเจ็กต์ ชนิดของคลาส ตลอดจนอธิบายถึงวิธีการสร้างคลาส สร้างออบเจ็กต์ และการเรียกใช้งานแอทริบิวต์และเมธอดในคลาส ซึ่งในการเขียนโปรแกรมเชิงวัตถุ คลาส เป็นส่วนที่ใช้อธิบายโครงสร้างของออบเจ็กต์ โดยออบเจ็กต์นั้นเป็นสิ่งที่จะต้องสร้างหรืออ้างอิงมาจากคลาส ภายในคลาสจึงประกอบด้วย แอทริบิวต์ และเมธอด ซึ่งเป็นองค์ประกอบที่สำคัญของออบเจ็กต์ ออบเจ็กต์ที่สร้างมาจากคลาสบางครั้งจะเรียกว่า อินสแตนท์ (instance of) ของคลาสนั้นๆ เมื่อสร้างออบเจ็กต์มาแล้ว เราสามารถเรียกใช้งานแอทริบิวต์และเมธอดของคลาสได้โดยใช้ออบเจ็กต์ตัวที่สร้างเป็นตัวอ้างอิงเพื่อเข้าถึงแอทริบิวต์และเมธอดที่ต้องการใช้งาน

3.8 เอกสารอ้างอิงบทที่ 3

กิตติ ภักดีวัฒนะกุล. (2551). Java เล่ม 1 (พิมพ์ครั้งที่ 7). กรุงเทพฯ : เคทีพี คอมพ์ แอนด์ คอนซัลท์

ธีรวัฒน์ ประกอบผล. (2554). คู่มือการเขียนโปรแกรมเชิงวัตถุภาษา JAVA OOP (ฉบับสมบูรณ์). กรุงเทพฯ : ซิมพลิฟาย.

อรพิน ประวัติบริสุทธิ์. (ม.ป.ป). คู่มือการเขียนโปรแกรมด้วยภาษา JAVA. กรุงเทพฯ : โปรวิชั่น จำกัด.

Farrell, J. (2010) Java Programming (5th ed.). Course Technology, Cengage Learning.

Oestereich, B. (2002). Developing Software with UML: Object–oriented analysis and design in practice. Pearson Education.

แบบฝึกหัดท้ายบทที่ 3

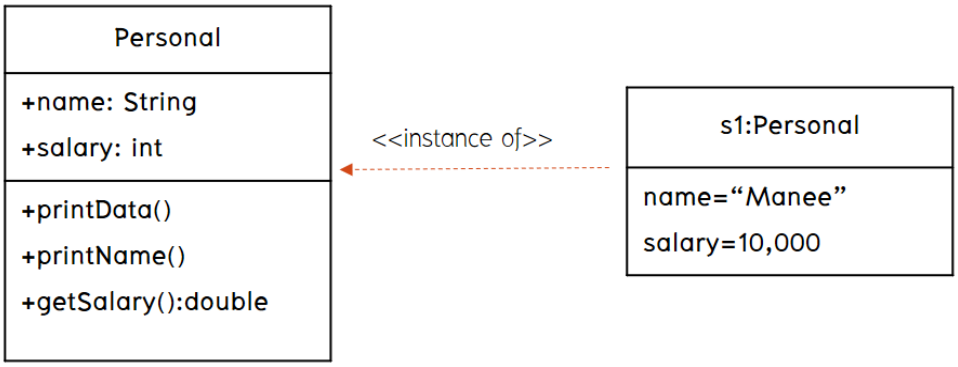
ตอนที่ 1 จงตอบคำถามต่อไปนี้

1. จงอธิบายความสัมพันธ์ระหว่างคลาส แอทริบิวต์ เมธอด และออบเจ็กต์

2. จงอธิบายความหมายและวิธีการใช้งาน Access Modifier แต่ละระดับ

3. การตั้งชื่อคลาส ชื่อว่า “My class” เป็นการตั้งชื่อถูกหรือผิด ถ้าผิดจงอธิบายถึงเหตุผลที่ผิดมาพอสังเขป

4. จงเขียนสร้างคลาสโปรแกรมลักษณะโครงสร้างตามแผนภาพด้านล่างนี้



5. จากคลาสในข้อ 4 หากต้องการเขียนคำสั่งสร้างออบเจ็กต์ และกำหนดให้ค่า name และ salary ของออบเจ็กต์ตามที่ระบุในแผนภาพความสัมพันธ์ระหว่างออบเจ็กต์และคลาส Personal จะต้องเขียนคำสั่งอย่างไร

โปรแกรมต่อไปนี้ใช้ตอบคำถามข้อ 6–7

```
public class PlusNumber{
        int x=0;        int y=0;        int z=0;
        public void printResult(){
                z=x+y;
                System.out.println(“The valute is ”+ z);
        }
}
```

6. หากต้องการเขียนคำสั่งกำหนดค่า x เท่ากับ 5 และ y เท่ากับ 15 ต้องเขียนคำสั่งอย่างไร

7. หากต้องการเรียกใช้งานเมธอด printResult() ต้องเขียนคำสั่งอย่างไร

8. จงบอกข้อผิดพลาดของโปรแกรมต่อไปนี้ พร้อมอธิบาย

```
public class Myclass();
        public int x;
        public double y, z;
        public void setNumber(int a, double b){
                x=a;            y=b;            z=a+b;
        }
}
```

9. จงบอกข้อผิดพลาดของโปรแกรมต่อไปนี้ พร้อมอธิบาย

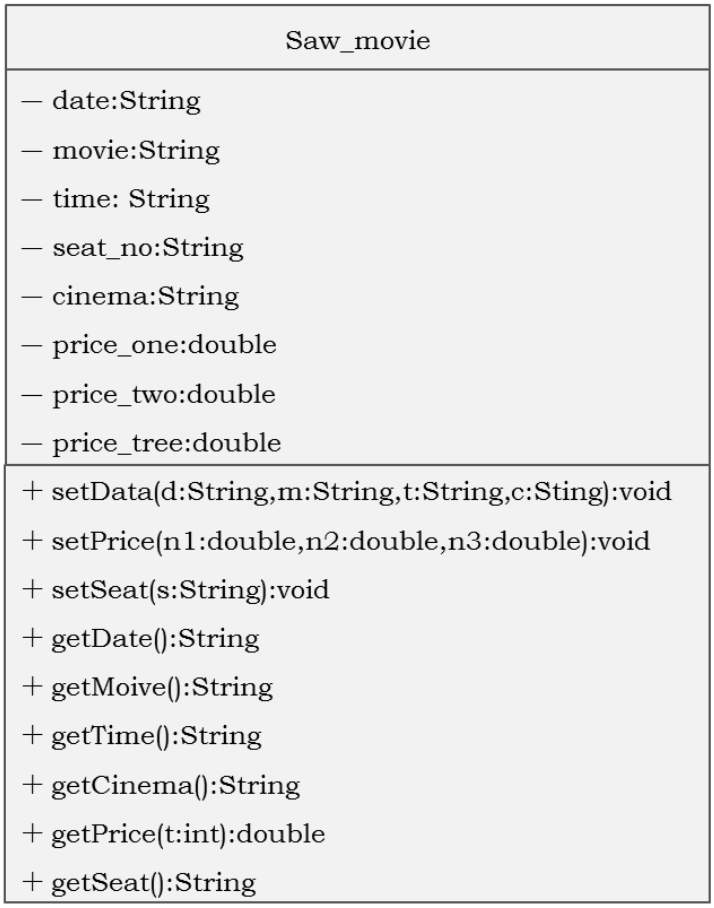
```
public class Myclass();
        public int x;
        public double y, z;
        public void setNumber(int a, double b){
                x=a+b;          y=b;            z=b*x;
        }
}
```

10. จงอธิบายการขั้นตอนการทำงานของโปรแกรมต่อไปนี้ พร้อมยกตัวอย่างผลลัพธ์

```
public class PlusNumber{
        int x=0;        int y=0;        int z=0;
        public static void main(String[] args) {
        PlusNumber obj= new PlusNumber ();
        Scanner key=new Scanner(System.in);
         obj.x=key.nextInt();
         obj.y=key.nextInt();
        obj.printResult();
         }
        public void printResult(){
                z=x+y;
                System.out.println("The valute is "+ z);
        }
}
```

ตอนที่ 2 จงเขียนโปรแกรมตามคำสั่งต่อไปนี้

1. จากคลาสไดอะแกรมต่อไปนี้ให้เขียนสร้างคลาสโปรแกรมให้ถูกต้อง



2. จากคลาสโปรแกรมในข้อ 1 ให้เขียนคำสั่งให้แต่ละเมธอด ทำงานดังนี้

- เมธอด setData() ใช้ในการกำหนดข้อมูล date, movie, time และ cinema
- เมธอด setPrice() ใช้ในการกำหนดข้อมูล price_one, price_two และ price_tree
- เมธอด setSeat() ใช้ในการกำหนดข้อมูล seat_no
- เมธอด getDate() ใช้อ่านค่าข้อมูล date
- เมธอด getMoive() ใช้อ่านค่าข้อมูล movie
- เมธอด getTime() ใช้อ่านค่าข้อมูล time
- เมธอด getCinema()ใช้อ่านค่าข้อมูล cinema
- เมธอด getPrice()ใช้ในการอ่านค่าข้อมูลราคา (price_one, price_two หรือ price_three
- เมธอด getSeat() ใช้ในการอ่านค่าข้อมูล seat_no

3. จงเขียนคำสั่งสร้างออบเจ็กต์ขึ้นมาจากคลาสที่สร้างในข้อ 1 และเขียนคำสั่งเรียกใช้งานเมธอดต่างๆ ภายในคลาส โดยมีรายละเอียดการใช้งานตามข้อความด้านล่างนี้

> *สมมติการซื้อตั๋วหนังเป็นการซื้อให้ตัวเองและเพื่อน จำนวน 2 ที่นั่ง โดยรายละเอียดการซื้อ คือ หนังเรื่อง "Pee-Mak" ฉายวันที่ "19-Sep-2017" รอบ 11:00-13:00 น. โรงที่ 3 ที่นั่ง A3 และ A4 ราคาตั๋วขึ้นอยู่กับประเภทของตั๋วที่เลือก โดยมีให้เลือกอยู่ 3 ประเภท แทนด้วยตัวเลข ดังนี้*
>
> *1 แทน ประเภท Internet Sofa ราคา 280 บาท*
>
> *2 แทน ประเภท Internet Pre ราคา 90 บาท*
>
> *3 แทน ประเภท Internet ราคา 70 บาท*

ให้เขียนคำสั่งแสดงผลลัพธ์ออกทางจอภาพ ตามรูปด้านล่างนี้

4. จงวิเคราะห์และออกแบบโปรแกรมสำหรับใช้รับค่าเลขจำนวนเต็มมา 5 ตัว แล้วให้โปรแกรมนับว่าเป็นเลขคู่กี่ตัว เป็นเลขคี่กี่ตัว

5. จงวิเคราะห์และออกแบบโปรแกรมสำหรับใช้ในการคำนวณหาพื้นที่และเส้นรอบวงของรูปสี่เหลี่ยม

ปฏิบัติการที่ 4
ฝึกการสร้างและใช้งานคลาส

วัตถุประสงค์

1. เพื่อให้ผู้เรียนสามารถสร้างและใช้งานคลาสได้

สาระสำคัญ

ในการเขียนโปรแกรมเชิงวัตถุนั้น จะเริ่มต้นจากการสร้างคลาส ดังนั้นผู้เขียนโปรแกรมก็จะต้องเข้าใจถึงวิธีการสร้างคลาสขึ้นมาใช้งาน แต่ก่อนจะสร้างคลาสได้นั้นผู้เขียนจะต้องเริ่มจากการจำแนกให้ได้ก่อนว่าอะไรคือ ออบเจ็กต์ที่เราสนใจ และออบเจ็กต์นั้นประกอบด้วยแอทริบิวต์และเมธอดอะไรบ้าง จากนั้นนำองค์ประกอบเหล่านั้นมารวมเป็นคลาส และสร้างคลาสขึ้นมาเป็นลำดับถัดไป

วิธีการ

1. ให้ผู้เรียนเปิดโปรแกรม NetBeans ขึ้นมา
2. ผู้สอนสาธิตวิธีปฏิบัติการสร้างและใช้งานคลาส และให้ผู้เรียนปฏิบัติตาม
3. ให้ผู้เรียนฝึกปฏิบัติสร้างและใช้งานคลาสด้วยตนเอง โดยใช้โจทย์จากแบบฝึกหัดท้ายบท ตอนที่ 2

บรรณานุกรม

ภาษาไทย

กิตติ ภักดีวัฒนะกุล. (2551). JAVA เล่ม 1 (พิมพ์ครั้งที่ 7). กรุงเทพฯ : เคทีพี คอมพ์ แอนด์ คอนซัลท์ จำกัด.

กิตติ ภักดีวัฒนะกุล และกิตติพงษ์ กลมกล่อม. (2547). UML วิเคราะห์และออกแบบระบบเชิงวัตถุ (พิมพ์ครั้งที่ 2). กรุงเทพฯ : บริษัท เคทีพี คอมพ์ แอนด์ คอนซัลท์ จำกัด.

ธีรวัฒน์ ประกอบผล. (2554). คู่มือการเขียนโปรแกรมเชิงวัตถุภาษา JAVA OOP (ฉบับสมบูรณ์). กรุงเทพฯ : ซิมพลิฟาย.

พนิดา พาณิชกุล. (2552). เทคโนโลยีเชิงวัตถุ (Object-Oriented technology). กรุงเทพฯ: เคทีพี คอมพ์ แอนด์ คอนซัลท์.

ภาษาจาวา. (2560, 29 มิถุนายน). ใน วิกิพีเดีย สารานุกรมเสรี. สืบค้นเมื่อ 20 ตุลาคม 2560, จาก https://th.wikipedia.org/wiki/ภาษาจาวา

วรรณวิภา ติตถะสิริ. (2558). การพัฒนาระบบสารสนเทศด้วยวิธีการเชิงวัตถุ. กรุงเทพฯ: เปเปอร์เมท (ประเทศไทย).

อรพิน ประวัติบริสุทธิ์. (ม.ป.ป). คู่มือการเขียนโปรแกรมด้วยภาษา JAVA. กรุงเทพฯ : โปรวิชั่น จำกัด.

โอภาส เอี่ยมสิริวงศ์. (2555). การวิเคราะห์และออกแบบระบบ (ฉบับปรับปรุงเพิ่มเติม). กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

MarcusCode. (สิงหาคม 20, 2559). String. สืบค้นเมื่อ 20 ตุลาคม 2560, จาก http://marcuscode.com/lang/java/string

Sakda Setrin. (กรกฎาคม 9, 2556). กฎการตั้งชื่อของภาษาจาวา. สืบค้นเมื่อ 20 ตุลาคม 2560, จาก http://java-sample-thai.blogspot.com/2013/07/blog-post.html

Sakda Setrin. (กรกฎาคม 13, 2556). Character Literal. สืบค้นเมื่อ 20 ตุลาคม 2560, จาก http://java-sample-thai.blogspot.com/2013/07/java-white-space.html

ภาษาอังกฤษ

Farrell, J. (2010). Java Programming (5th ed.). Course Technology, Cengage Learning.

Java SE Downloads. (n.d.). Retrieved October 20, 2017, from www.oracle.com

NetBeans IDE. (n.d.). Retrieved October 20, 2017, from https://netbeans.org

Oestereich, B. (2002). Developing Software with UML: Object-oriented analysis and design in practice. Pearson Education.

Oracle. (n.d.) Class String. Retrieved October 20, 2017, from
https://docs.oracle.com/javase/7/docs/api/java/lang/String.html