

วัตถุประสงค์

- A. เพื่อเข้าใจโครงสร้างของคลาสสำหรับการสร้างออบเจกต์ และการเรียกใช้เมธอด
- B. เพื่อเข้าใจ constructor
- D. ทบทวนการใช้ ArrayList
- C. เพื่อเข้าใจ method overloading
- E. เพื่อเข้าใจ reference
- F. เพื่อเข้าใจปฏิสัมพันธ์ของออบเจกต์

กิจกรรมที่ 1

1.1 เขียน Lab1Coder.java ตาม diagram ประกอบ (ยังไม่ต้องทำส่วนสีเทา)

1.2 เขียน Lab1Driver.java

1.3 สั่ง javac Lab1Driver.java (แก้ error จน compile ให้สำเร็จ)

1.4 เมื่อ compile ผ่านได้ไฟล์อะไรเพิ่มขึ้นมาบ้าง Lab1Coder.class, Lab1Driver.class

1.5 ต้อง compile Lab1Coder.java ต่างหากหรือไม่ X, No, ไม่, false

```

3 public class Lab1Driver {
4     // static Lab1Coder keng;
5     // static Lab1Coder somsri;
6
7     public static void main(String[] args) {
8         q1();
9     }
10
11     static void q1() {
12         Lab1Coder santa = new Lab1Coder();
13         santa.setName(n: "Santa");
14         santa.setExperience(exp: 3);
15         String str = String.format(format: "%s has been working for %d years.",
16                                     , santa.getName(), santa.getExperience());
17         System.out.println(str); //Santa has been working for 3 years.
18     }
19 }

```

Lab1Coder

– name : String *private*
 experience : int
 – languages : ArrayList<String>

Lab1Coder(n : String, exp : int) :

setName(name : String) : void

getName() : String

setExperience(exp : int) : void

getExperience() : int

setLanguages(String ... lang): void

getLanguages() : ArrayList<String>

+ toString() : String

findCommonLanguages(Lab1Coder other) : ArrayList<String>

1.6 เหตุที่ จาวา compile

Lab1Coder.java ให้เพราะ ถูกเรียกใช้ โดย Lab1Driver

1.7 เรียก Lab1Driver ให้

ทำงาน ใช้คำสั่ง java Lab1Driver

1.8 สัญลักษณ์ – แสดง access modifier อะไร private

1.9 ใน q1 เรา access santa.name ได้หรือไม่ (เช่น `println(santa.name)`) ไม่ได้
อยู่

1.10 ใน q1 เรา access santa.experience ได้หรือไม่ (เช่น `println(santa.experience)`) ได้

1.11 ใน q1 เรา modify santa.name ได้หรือไม่ (เช่น `santa.name = "claus";`) ไม่ได้

1.12 ใน q1 เรา modify santa.experience ได้หรือไม่ (เช่น `santa.experience = 99;`) ได้

1.13 เข้าใจผลของการกำกับด้วย private หรือไม่ เข้าใจ



1.14 ใน `q1()` ใช้ไวยากรณ์อะไรในการสร้าง object santa `new Lab1Coder();`

1.15 ใน `main` เห็น santa หรือไม่ ไม่

1.16 ลบ `Lab1Coder.class` แก้ไข `class Lab1Coder {` เป็น `class Xab1Coder` บันทึก `Lab1Coder.java` แล้วสั่ง `javac Lab1Coder.java` (ไม่ต้องแก้ชื่อไฟล์) จะคอมไพล์ผ่านหรือไม่ ผ่าน ได้ไฟล์ชื่ออะไร `Xab1Coder.class`

1.17 รัน `Lab1Driver` แล้ว error พยายามสื่อว่าอะไร หา class Lab1Coder ไม่เจอ เนื่องจากเรา compile เป็น `Xab1Coder` เพื่อ compile จะได้ไฟล์ออกมาตามชื่อ class

1.18 ลบ `Xab1Coder.class` กลับไปแก้คือให้เป็น `class Lab1Coder` แล้วบันทึกไฟล์

กิจกรรมที่ 2

2.1 เขียน Lab1Coder(String n, int exp)
{ } พร้อม new ArrayList<>(); ให้
languages

2.2 เขียน setLanguage(String ...lang)
(lang เป็น array of String)

2.3 เขียน Lab1Coder() { } เพื่อให้ยัง
สามารถเรียก q1() ได้

2.4 เหตุใดต้องสร้าง Lab1Coder() { } ทั้งที่
กิจกรรมที่ 1 ไม่ต้องสร้าง

เพื่อให้ q1 เรียก func โดยไม่ต้องใส่
param ได้

2.5 เรียกเทคนิคการสร้าง method ที่ชื่อ
เหมือนกัน แต่ signature ไม่เหมือนกันว่า
Overloading

หมายเหตุ เนื่องจากเราจะใช้ somsri ที่
method อื่น จึงประกาศไว้เป็น class
variable

Lab1Coder
– name : String experience : int – languages : ArrayList<String>
Lab1Coder(n : String, exp : int) : setName(name : String) : void getName() : String setExperience(exp : int) : void getExperience() : int setLanguages(String ... lang): void getLanguages() : ArrayList<String> + toString() : String findCommonLanguages(Lab1Coder other) : ArrayList<String>

```

3 public class Lab1Driver {
4     static Lab1Coder keng;
5     static Lab1Coder somsri;
6
7     Run | Debug
8     public static void main(String[] args) {
9         // q1();
10        q2_properConstructor();
11    }
12
13    // static void q1() {
14    //     Lab1Coder santa = new Lab1Coder();
15    //     santa.setName("Santa");
16    //     santa.setExperience(3);
17    //     String str = String.format("%s has been working for %d years.",
18    //     santa.getName(), santa.getExperience());
19    //     System.out.println(str); // Santa has been working for 3 years.
20    // }
21
22    static void q2_properConstructor() {
23        somsri = new Lab1Coder(n: "Somsri", exp: 5);
24        somsri.setLanguages(...lang: "javascript", "dart");
25        System.out.println(somsri.getLanguages()); //[[javascript, dart]]
26    }
27 }

```

กิจกรรมที่ 3

3.1 เขียน Lab1Coder(String n) {} โดยให้ experience เป็น 0

3.2 เขียน public String toString() {} ตามตัวอย่าง output ที่แสดง

3.3 เขียน findCommonLanguages(Lab1Coder other) โดยหากทั้ง 2 คนไม่รู้จักภาษาเดียวกันสักรายภาษาให้ตอบ none

```
Santa has been working for 3 years.
[javascript, dart]
ber3(0) knows c java typescript
[none]
[java, c]
```

Lab1Coder
- name : String experience : int - languages : ArrayList<String>
Lab1Coder(n : String, exp : int) : setName(name : String) : void getName() : String setExperience(exp : int) : void getExperience() : int setLanguages(String ... lang): void getLanguages() : ArrayList<String> + toString() : String findCommonLanguages(Lab1Coder other) : ArrayList<String> Lab1Coder(n : String) :

```
28 static void q3() {
29     keng = new Lab1Coder(n: "Keng", exp: 2);
30     keng.setLanguages(...lang: "java", "solidity", "c");
31     Lab1Coder ber3 = new Lab1Coder(n: "ber3");
32     ber3.setLanguages(...lang: "c", "java", "typescript");
33     System.out.println(ber3);
34
35     ArrayList<String> commonLanguages = ber3.findCommonLanguage(somsri);
36     System.out.println(commonLanguages);
37     commonLanguages = keng.findCommonLanguage(ber3);
38     System.out.println(commonLanguages);
39 }
```

สรุปหลักการ encapsulation พอสังเขป

เป็นการปกปิด object ให้สามารถเข้าถึงได้เฉพาะส่วนที่กำหนด
 ส่วนในยูนิคใช้ร่วมกับ object ภายนอกเพื่อไม่ให้สามารถเข้าถึงข้อมูลได้โดยตรง เช่น การคำนวณภาษีที่เราไม่จำเป็นต้อง
 เปิดหน่วย ส่วน การคำนวณ เราได้แค่ผลลัพธ์ก็เพียงพอแล้ว
 tax fee (ร.) → \$

กำหนดส่ง TBA