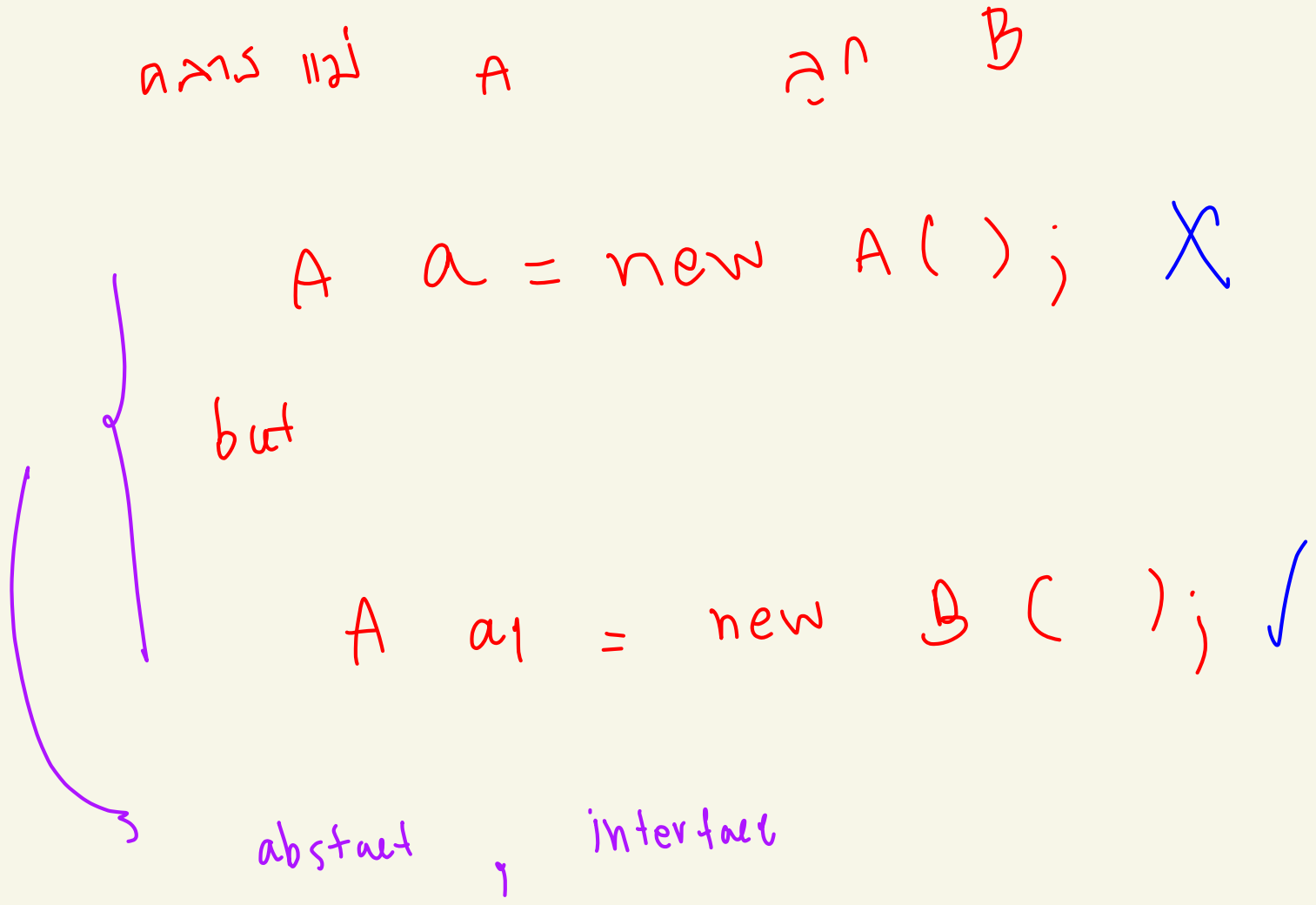
extends คลาสลูก มี method เหมือน
คงาสแม่ แต่เพิ่มตัวข้างในได้

ใช้คำสั่ง super สำหรับเรียกสิ่งที่อยู่ใน
คลาสแม่

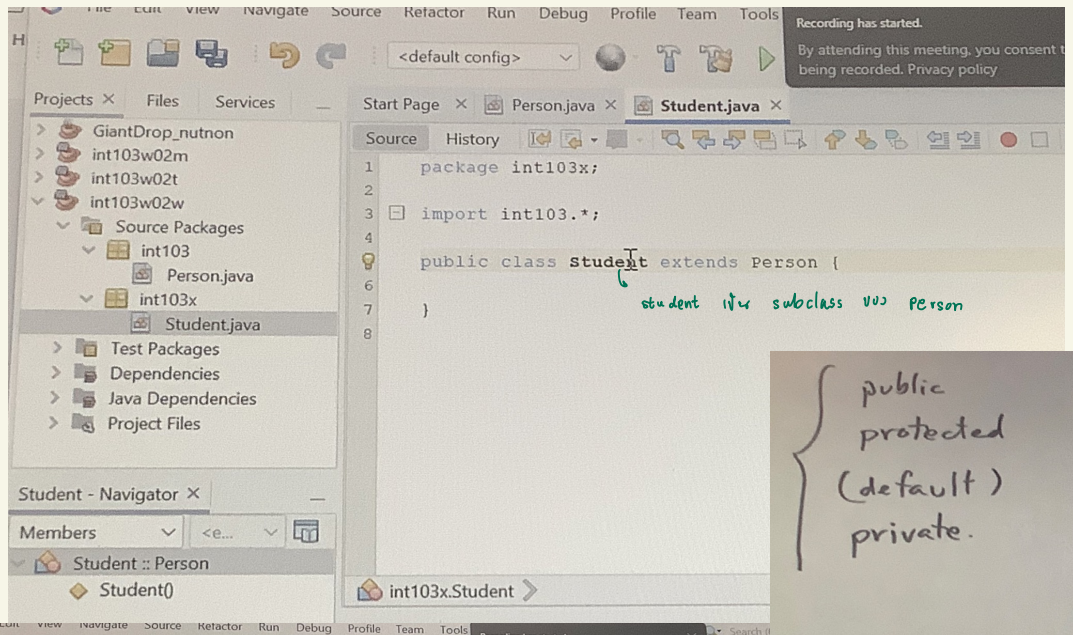
abstract คือ ประเภทคลาสแม่ ที่ทำ
โครง method ไว้ แต่จะไม่มีไส้ข้างใน
ให้ลูกไปเติมเอง

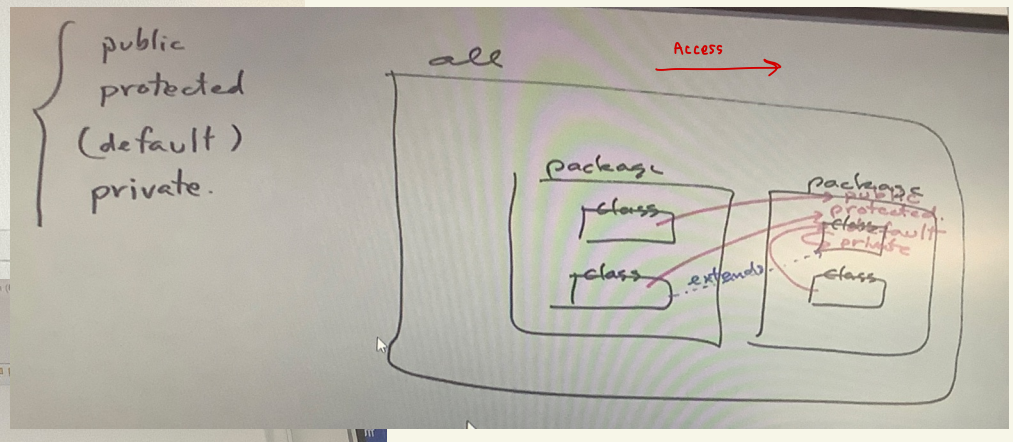
interface แม่ มีลูกคือ implement
จะคล้ายๆ abstract และ
ลูก มีแม่ได้หลายคน แตกต่างจาก
inheritance ทำมะดาที่ลูกมีแม่ได้แค่คน
เดียว

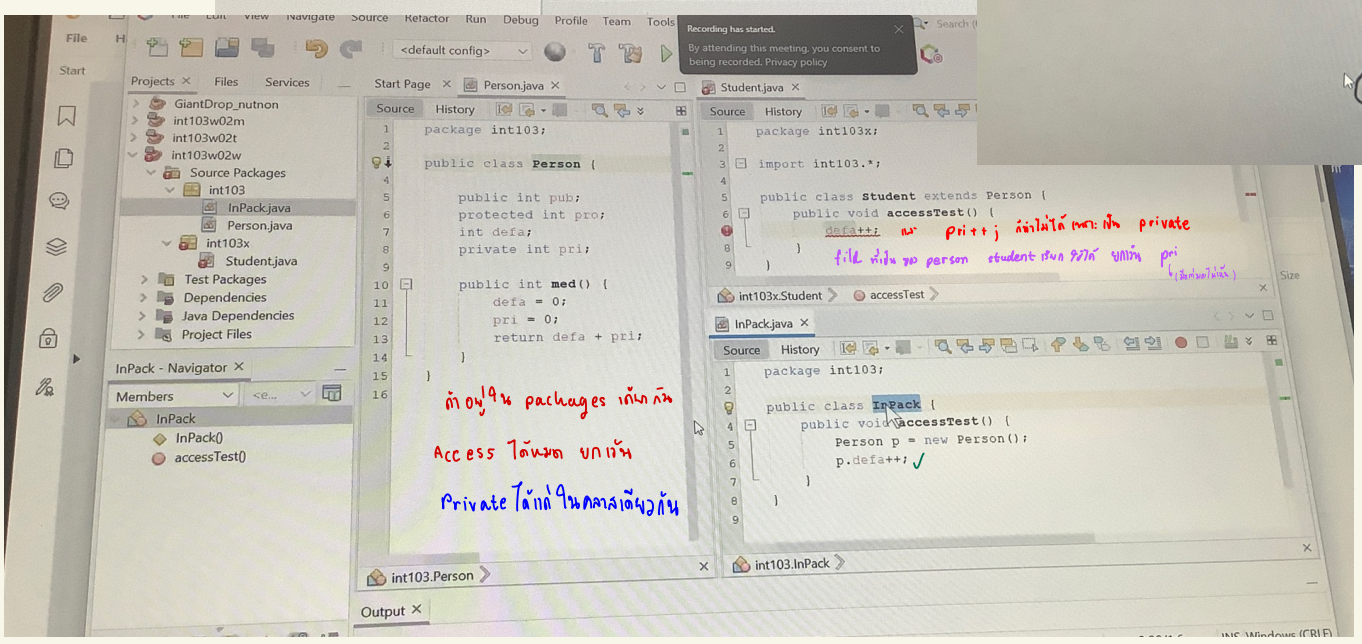
ลูกใช้ method คลาสแม่ได้ แต่ต้องมี
@Override

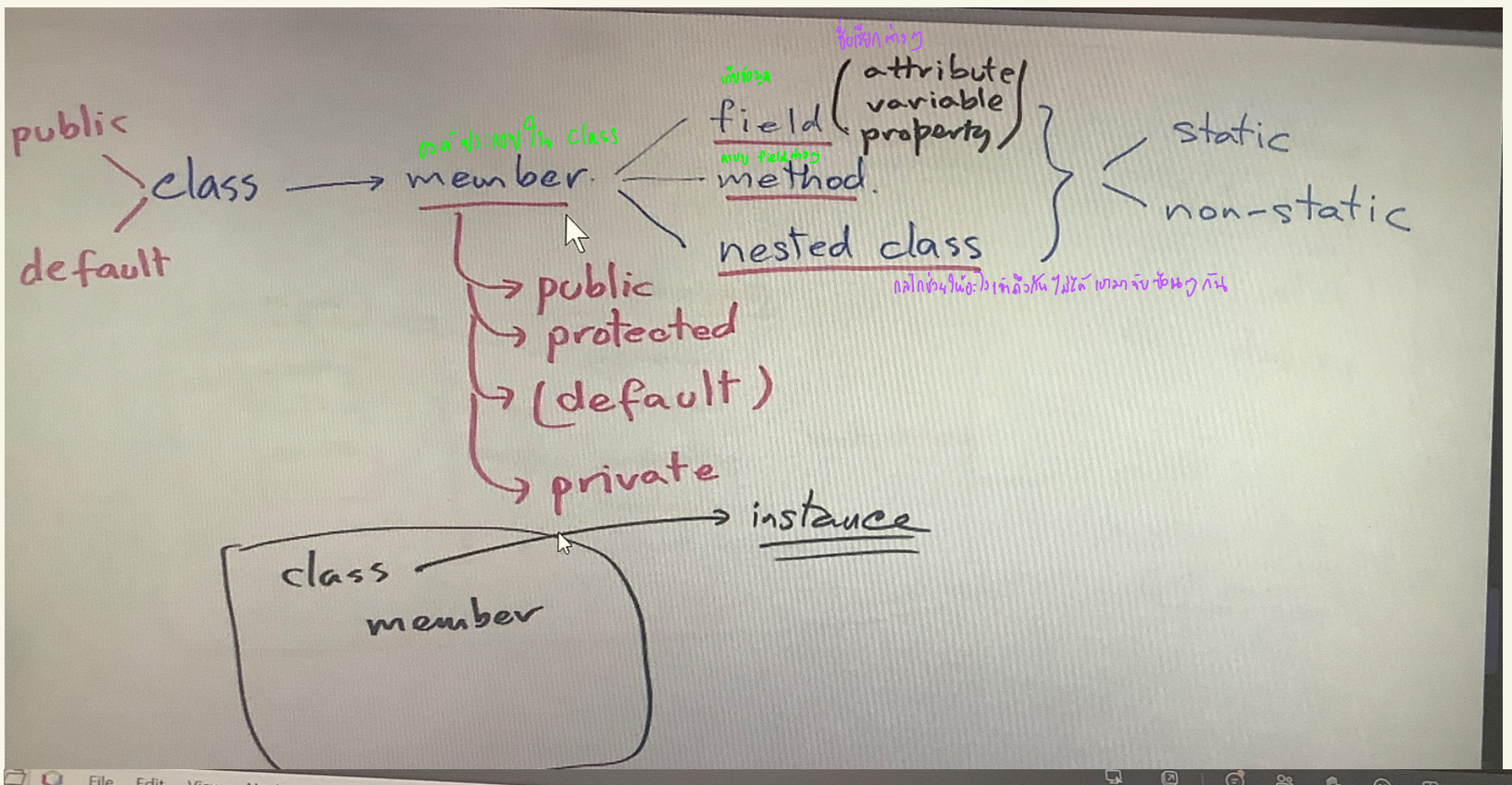


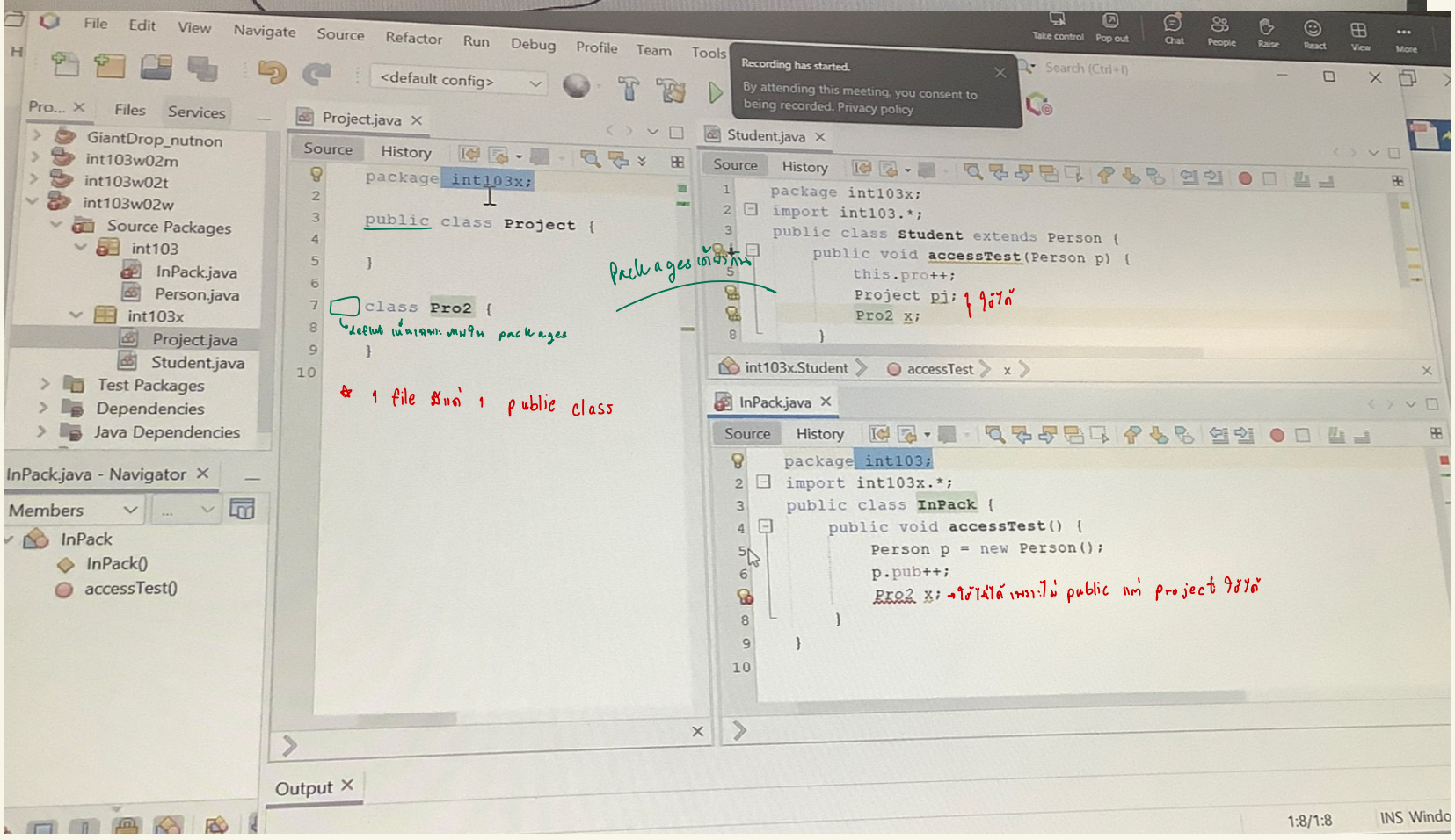
Encapsulation



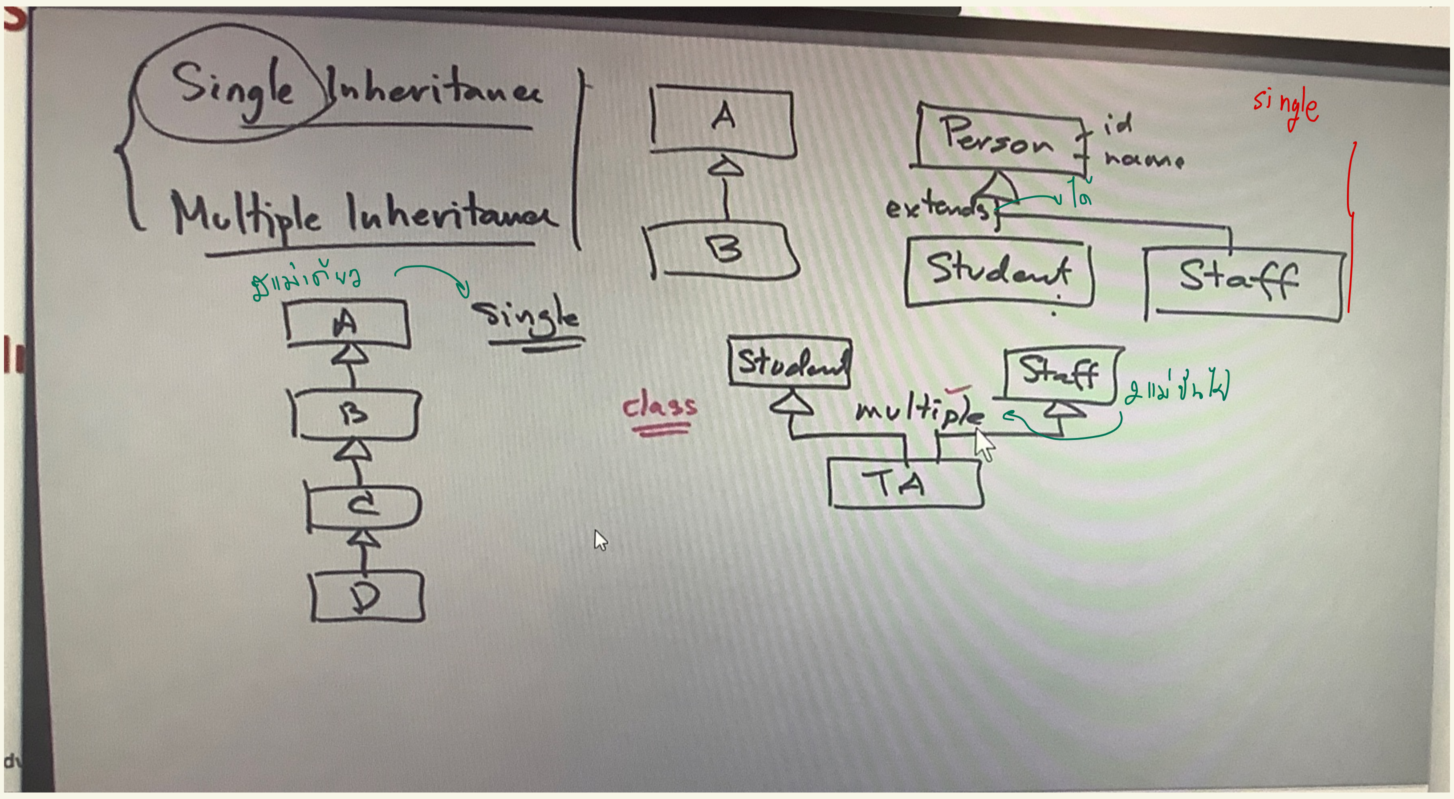




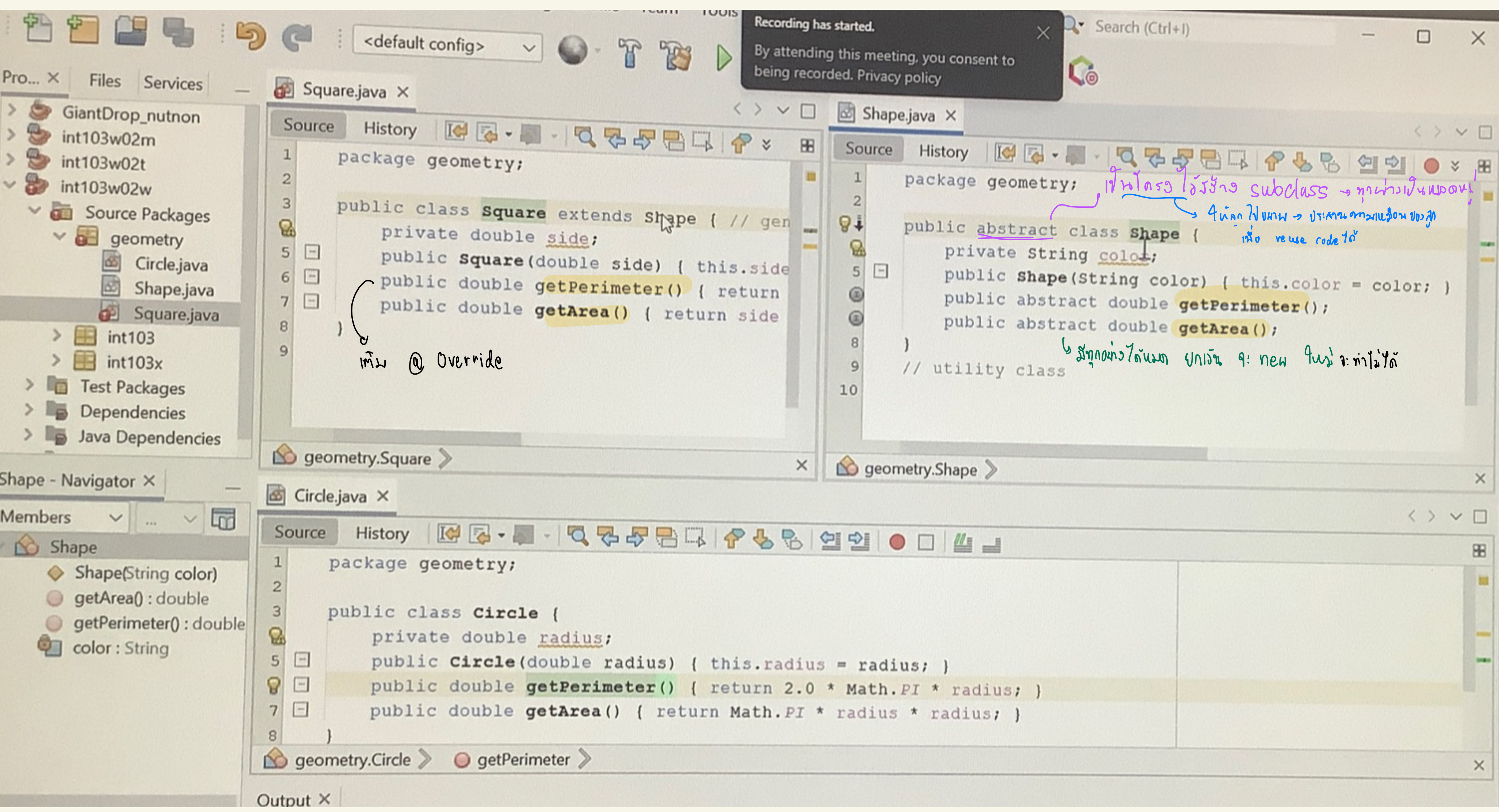




inheritane



abstract → ไม่มี body → ไม่รู้ว่า มันคำนวณอย่างไร



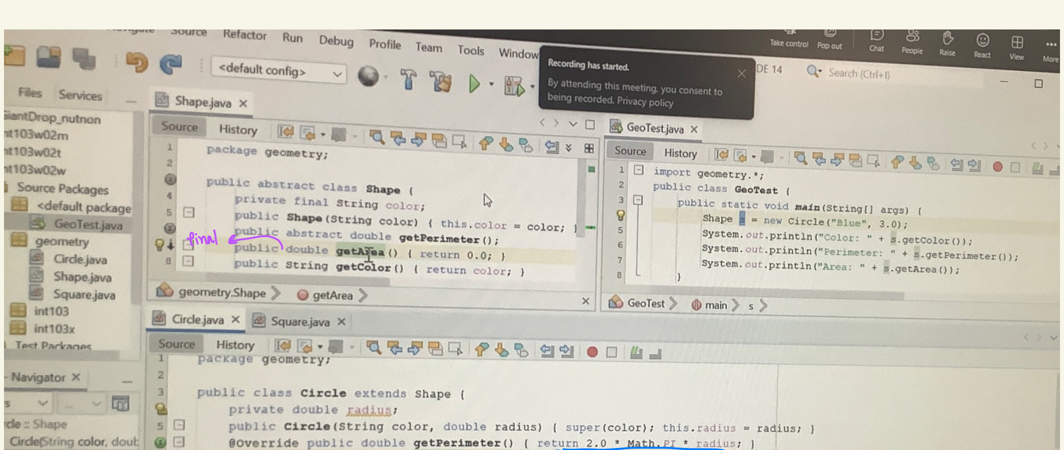
square เป็น shape ถ้าสร้าง square จะสร้าง shape จากนั้น call super ( ) ให้ จาก side ของ square

public shape (color) ; this . side = side;

เรียก constructor ของ super class (class แม่)

super ( ) เรียก empty constructor or default constructor

★ ทำ super class ก่อน → subclass ทีหลัง



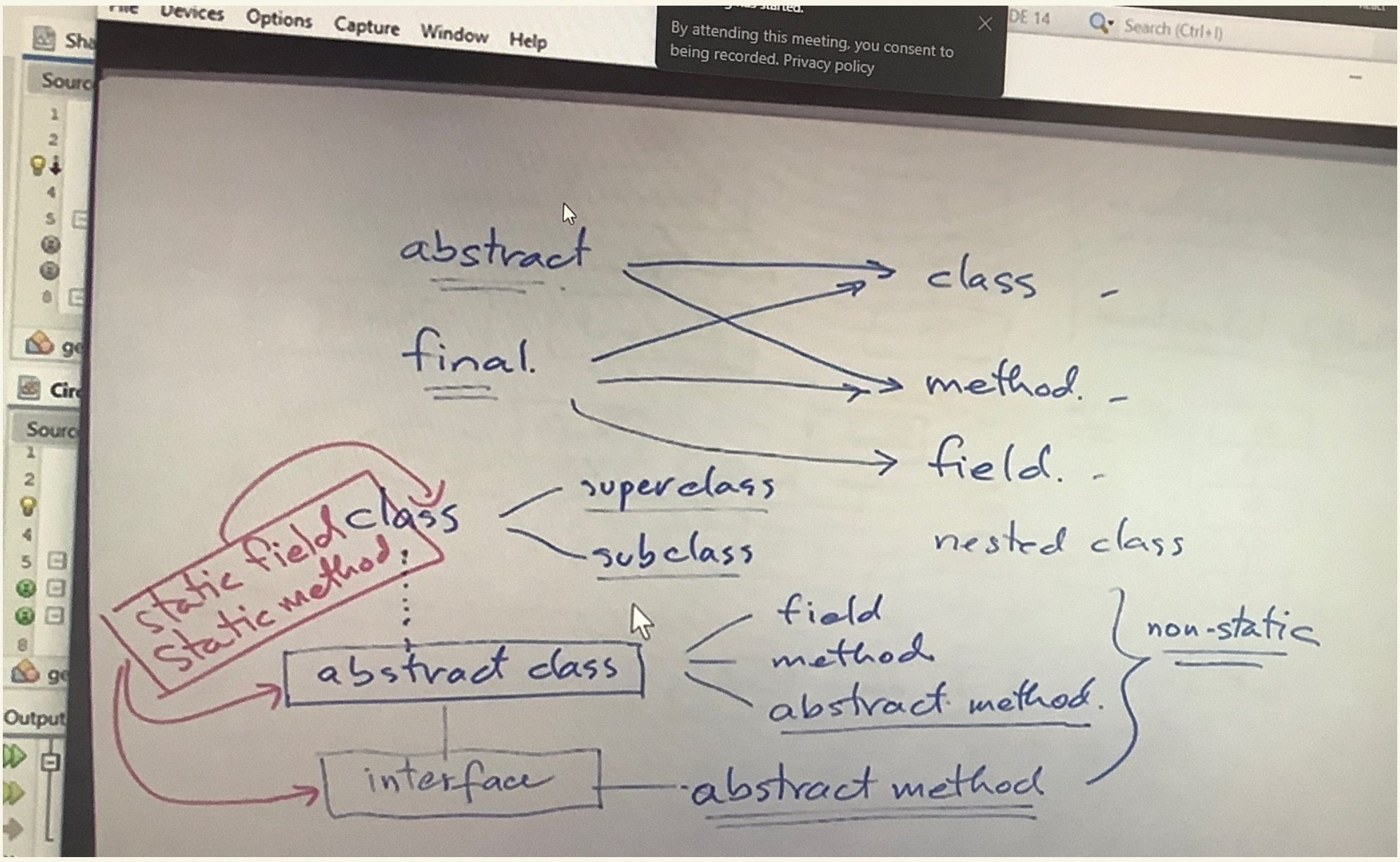
ถ้าเป็น final จะ ใหม่ไม่ได้
ต้องใช้แบบ ตัว แม่ เท่านั้น

(non-static) abstract ตรงข้ามกับ final อยู่ด้วยกันไม่ได้

- abstract medthod
  - ไม่มี body → ให้ลูกขยาย
  - มี subclass ได้
  - field เป็น abstract ไม่ได้
- final medthod
  - มี body ไม่ให้ลูกไปขยายต่อ
  - มี sub class ไม่ได้
  - field เป็น final ได้ – ใส่ค่าแล้ว เปลี่ยน ไม่ได้

Interface

- คล้าย abstract class → non static
  - ข้อดี คือ abstract method ตัวแรกเลย → non static
  - ทำให้ เกิด multiple inheritance ได้



ตอน static มีข้อแตกต่าง → Utility

being recorded. Privacy policy

abstract class → class. ที่มีไว้ สร้าง object จาก class นี้ไม่ได้.

Tiger / Lion / Cat → implement

interface → ภายในจะมีแต่ non-static member ← field, method.

จาก multiple inheritance กับ interface เหมือน

concrete class
มี { field ได้
method.
method ทุกตัว ต้องมี body (มี implementation)

มีก็ไม่มี implementation ก็ได้

abstract class
มี { field. ได้
method
abstract method.
- มี method ธรรมดา มี body ได้
- หรือ อาจมี body บาง method ก็ได้
- หรือ จะไม่มี body เลยสักอัน method ก็ได้

interface
ตัวแปรมีแต่ field.
มี method ได้ abstract
ไม่มี method ใดมี implementation เลย

USB2.0 VISUALIZER

interface

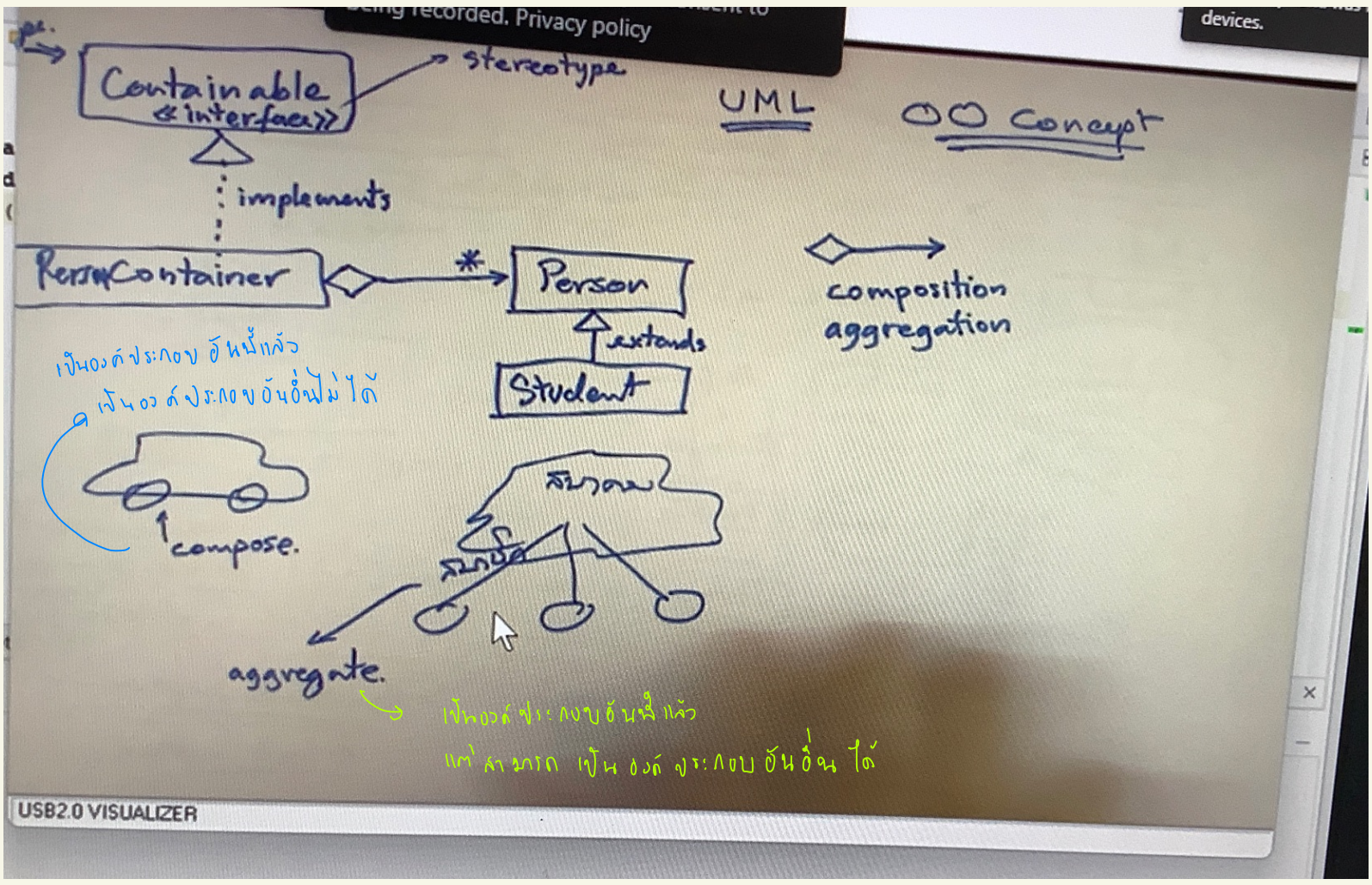
```
Private  Person [ ] box = new Person [ size ];

Public   boolean add (Object o)

    if (!(o instanceof Person)) return false;
```

↳ ถ้า o ไม่เป็น Person ให้ false
ไม่งั้น ถ้าเป็น instance ของ Person
o instanceof Person => o เป็น object ของ Person / subclass ของ Person

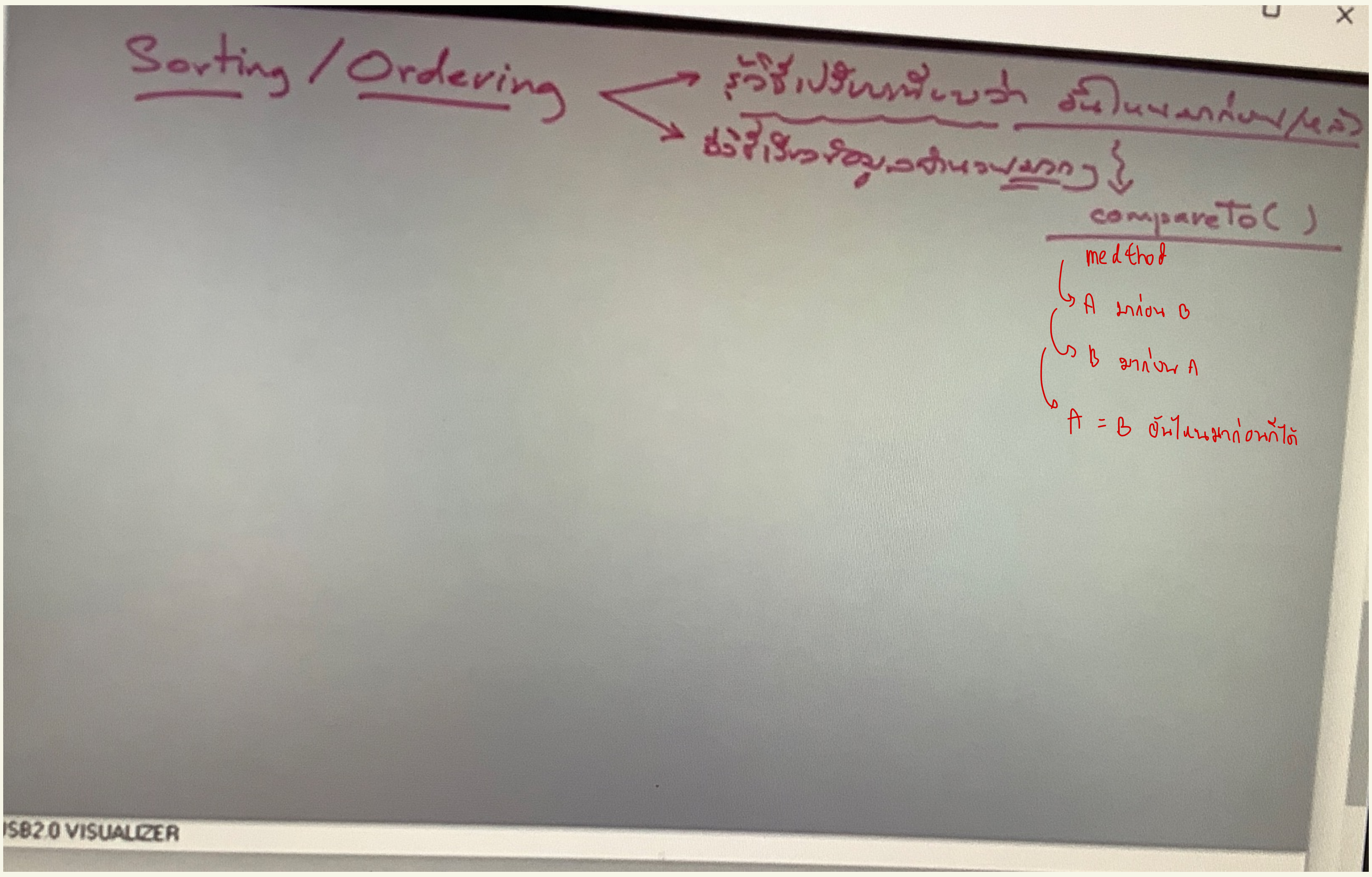
```
1  package containment;
2
3  public interface Containable {
4      public boolean add(Object o);
5      public boolean find(Object o);
6  }
```

add(Object o); → ใส่ object
find(Object o); → หา object
return true หาเจอ
false หาไม่เจอ

being recorded. Privacy policy



USB2.0 VISUALIZER



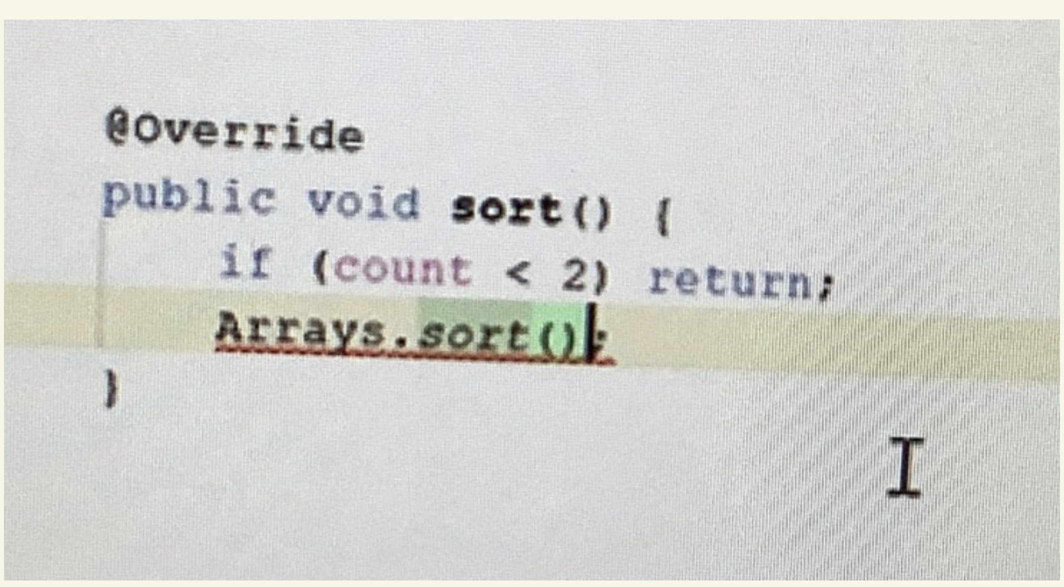
int i → i เป็น primitive • มี math . ________ ฟังก์ชัน ต่าง

Interger i → i เป็น object

Array ไม่มี class ที่ชื่อว่า Array → ไม่มี medthod

↳ Arrays. ____

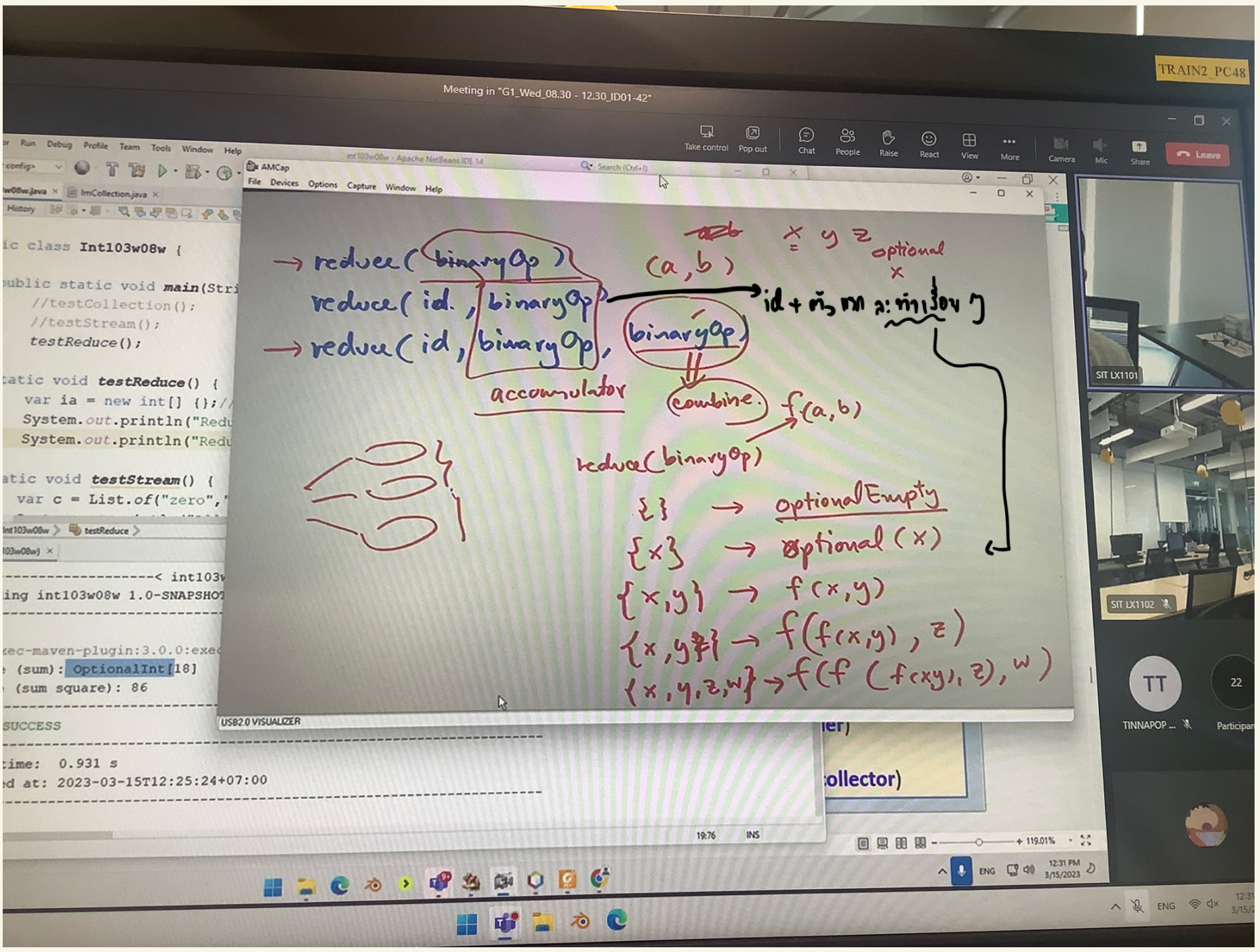
Arrays. ____ (   )



```
@Override
public void sort() {
    if (count < 2) return;
    Arrays.sort();
}
```

```java
public class Int103w08w {

    public static void main(String[] args) {
        //testCollection();
        testStream();
    }
    static void testStream() {
        var c = List.of("zero","one","two","three","four","five","six","seven");
        System.out.println("All < 10: " + c.stream().allMatch(s->s.length()<10));
        System.out.println("Any < 4: " + c.stream().anyMatch(s->s.length()<4));
        System.out.println("None < 3: " + c.stream().noneMatch(s->s.length()<3));
        List<String> e = List.of();
        System.out.println("All: " + e.stream().allMatch(s->false));
        System.out.println("None: " + e.stream().noneMatch(s->false));
    }
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ int103w08w ---
All < 10: true
Any < 4: true
None < 3: true
All: true
None: true
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.949 s
Finished at: 2023-03-15T11:42:47+07:00
------------------------------------------------------------------------
```



Compare To

String

ค่าติดลบ (negative value) หาก a < b

ศูนย์ (zero value) หาก a = b

ค่าบวก (positive value) หาก a > b

```
CompareToDemo > src > com > prasertcbs > Main
Main.java
1  package com.prasertcbs;
2
3  public class Main {
4
5      public static void main(String[] args) {
6          compareString("apple", "coconut");
7          compareString("apple", "apple");
8          compareString("coconut", "apple");
9
10     }
11
12     public static void compareString(String a, String b) {
13         System.out.printf("a = %s, b = %s, a.compareTo(b) = %d%n", a, b, a.compareTo(b));
14     }
15
16 }
17
Run Main
"C:\Program ...
a = apple, b = coconut, a.compareTo(b) = -2a = apple, b = apple, a.compareTo(b) = 0a = coconut,
Process finished with exit code 0
```

ตัวเลข

```
CompareToDemo > src > com > prasertcbs > Main
Main.java
3  public class Main {
4
5      public static void main(String[] args) {
6          compareString("apple", "banana");
7          compareString("apple", "apple");
8          compareString("coconut", "apple");
9
10     }
11
12     public static void compareString(String a, String b) {
13         System.out.printf("a = %s, b = %s, a.compareTo(b) = %d%n", a, b, a.compareTo(b));
14     }
15
16     public static void compareIntegerObject(Integer a, Integer b) {
17         System.out.printf("a = %s, b = %s, a.compareTo(b) = %d%n", a, b, a.compareTo(b));
18     }
19
Run Main
"C:\Program ...
a = apple, b = banana, a.compareTo(b) = -1
a = apple, b = apple, a.compareTo(b) = 0
a = coconut, b = apple, a.compareTo(b) = 2
Process finished with exit code 0
```

Integer เป็น Wrapper class เพื่อแปลง int ให้เป็น Integer Object

ค่า -1 หาก a < b

ค่า 0 หาก a = b

ค่า 1 หาก a > b

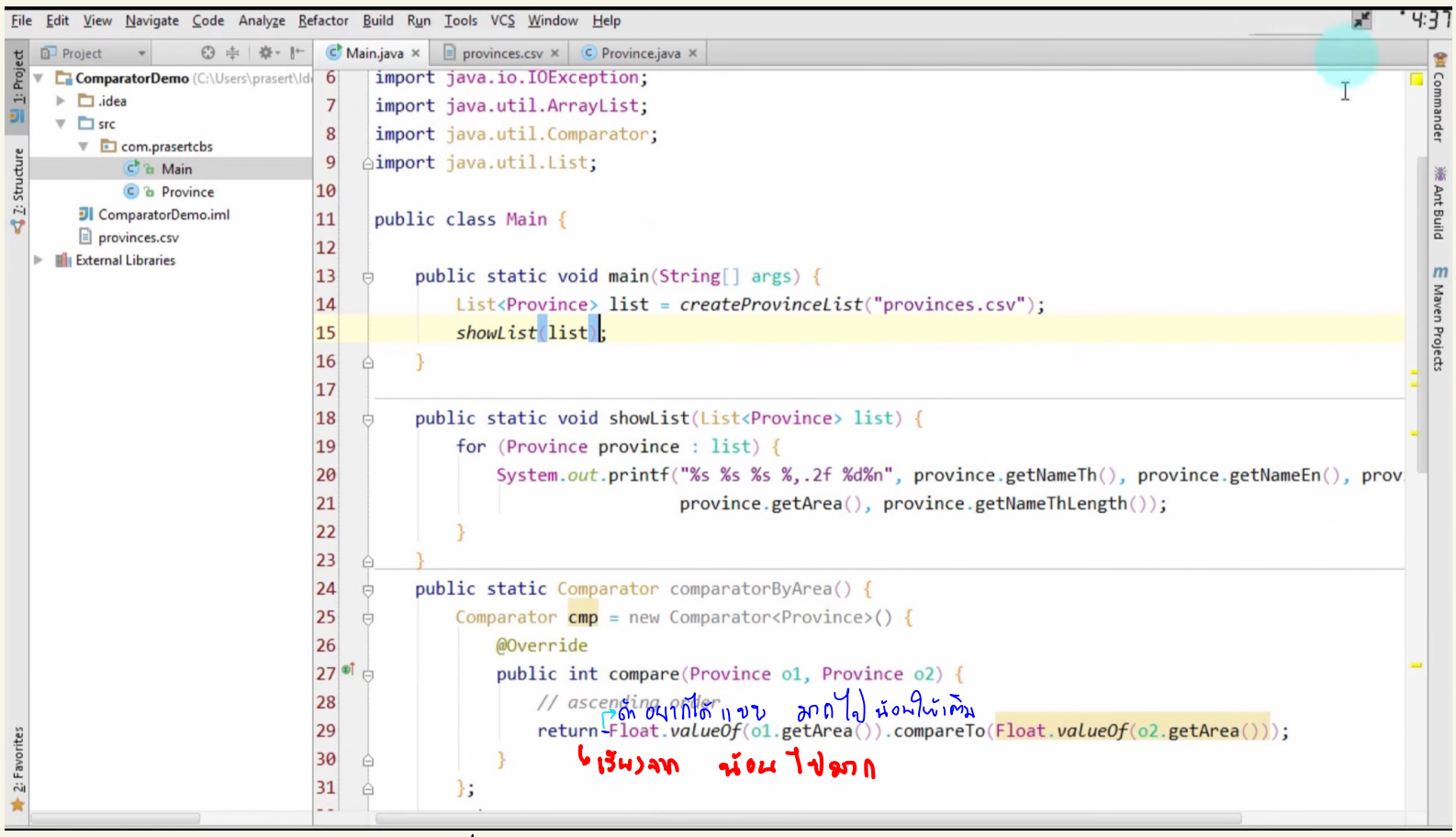
```
a = coconut, b = apple, a.compareTo(b) = 2
```

ก่อนหน้านี้ primitive

```
CompareToDemo > src > com > prasertcbs > Main
Main.java
16         compareIntegerPrimitive(7, 5);
17
18     }
19
20     public static void compareString(String a, String b) {
21         System.out.printf("a = %s, b = %s, a.compareTo(b) = %d%n", a, b, a.compareTo(b));
22     }
23
24     public static void compareIntegerObject(Integer a, Integer b) {
25         System.out.printf("a = %d, b = %d, a.compareTo(b) = %d%n", a, b, a.compareTo(b));
26     }
27
28     public static void compareIntegerPrimitive(int a, int b) {
29         System.out.printf("a = %d, b = %d, a.compareTo(b) = %d%n", a, b,
30                 Integer.valueOf(a).compareTo(Integer.valueOf(b)));
31     }
32 }
33
Run Main
a = coconut, b = apple, a.compareTo(b) = 2
a = 5, b = 7, a.compareTo(b) = -1
a = 5, b = 5, a.compareTo(b) = 0
a = 7, b = 5, a.compareTo(b) = 1

Process finished with exit code 0
```

```
CompareToDemo > src > com > prasertcbs > Main
Main.java
35
36         System.out.printf("a = %d, b = %d, a - b = %d%n", a, b,
37                 a - b);
38     }
39
40     public static void compareFloatPrimitive(float a, float b) {
41         System.out.printf("a = %.2f, b = %.2f, a.compareTo(b) = %d%n", a, b,
42                 Float.valueOf(a).compareTo(Float.valueOf(b)));
43     }
44 }
45
Run Main
"C:\Program ...
a = apple, b = banana, a.compareTo(b) = -1
a = apple, b = apple, a.compareTo(b) = 0
a = coconut, b = apple, a.compareTo(b) = 2
a = 7.50, b = 5.12, a.compareTo(b) = 1
a = 7.50, b = 7.50, a.compareTo(b) = 0
a = 5.12, b = 7.50, a.compareTo(b) = -1

Process finished with exit code 0
```

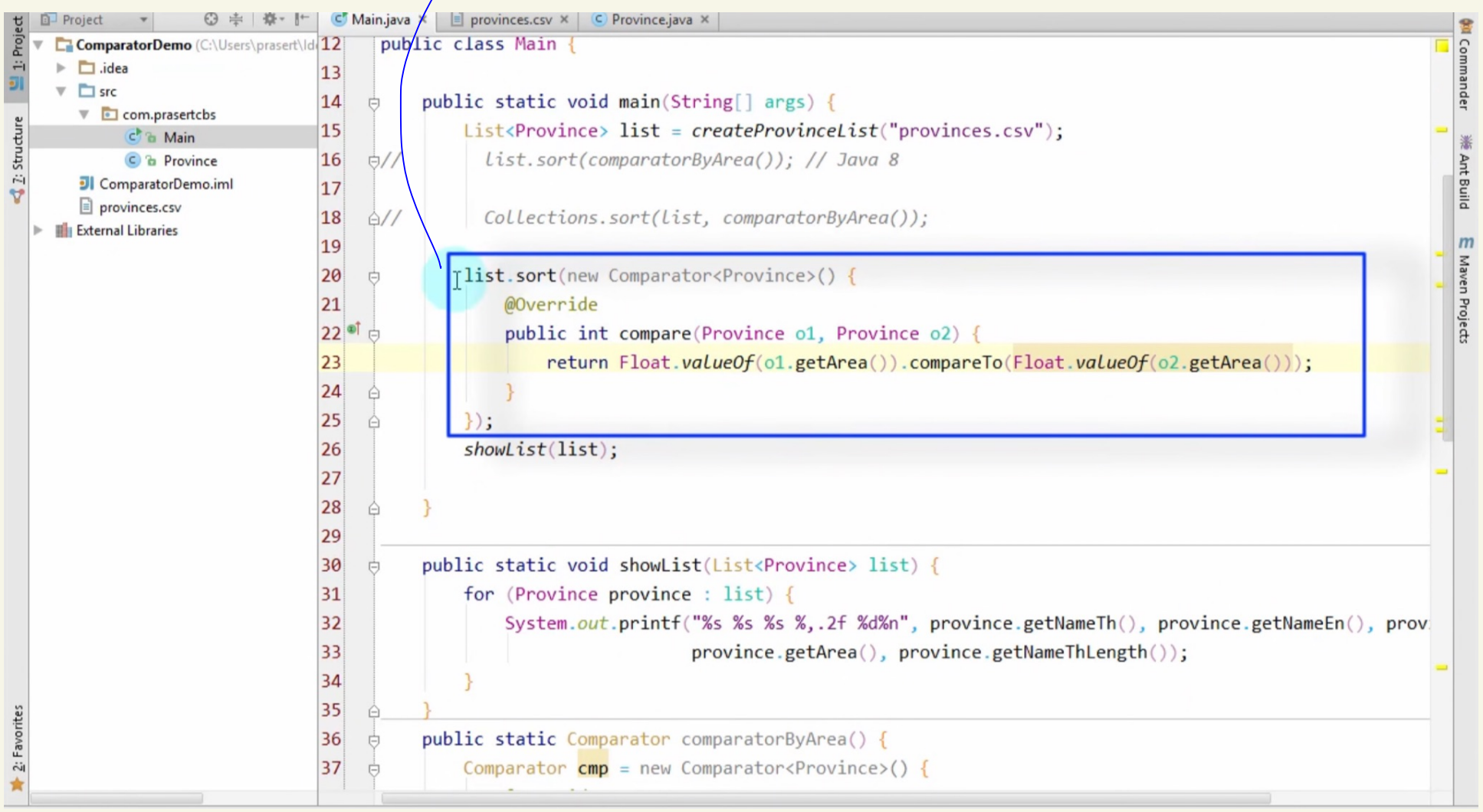
Comparator เรียง ArrayList



ลืม เขียนคำสั่งนี้

```
    public static void main(String[] args) {
        List<Province> list = createProvinceList("provinces.csv");
//      list.sort(comparatorByArea()); // Java 8

//      Collections.sort(list, comparatorByArea());
```

เขียนแบบนี้ก็ได้



```
//        list.sort(new Comparator<Province>() {
//            @Override
//            public int compare(Province o1, Province o2) {
//                return o1.getNameEn().compareTo(o2.getNameEn());
//            }
//        });

        list.sort(new Comparator<Province>() {
            @Override
            public int compare(Province o1, Province o2) {
                Collator coll = Collator.getInstance(new Locale("th", "TH"));
                return coll.compare(o1.getNameTh(), o2.getNameTh());
//                return o1.getNameTh().compareTo(o2.getNameTh());
            }
        });
```

เรียงตามอักษรอังกฤษ

เรียงแบบคำอ่านไทย

แบบ Unicode

```
อ่างทอง Ang Thong อท 968.37 7
เชียงราย Chiang Rai ชร 11,678.37 8
เชียงใหม่ Chiang Mai ชม 20,107.06 9
เพชรบุรี Phetchaburi พบ 6,225.14 8
เพชรบูรณ์ Phetchabun พช 12,668.42 9
เลย Loei ลย 11,424.61 3
แพร่ Phrae พร 6,538.60 4
แม่ฮ่องสอน Mae Hong Son มส 12,681.26 10

Process finished with exit code 0
```

```
        Collator coll = Collator.getInstance(new Locale("th", "TH"));
        return coll.compare(o1.getNameTh(), o2.getNameTh());
//          return o1.getNameTh().compareTo(o2.getNameTh());
    }
);

t.sort(new Comparator<Province>() {
  @Override
  public int compare(Province o1, Province o2) {
      return o1.getNameThLength() - o2.getNameThLength();
//        return Integer.valueOf(o1.getNameThLength()).compareTo(Integer.valueOf(o2.getNameThLength()));
  }
```

เรียงตัวอักษร นครน้อย → มาก

```
ชลบุรี Chon Buri ชบ 4,611.83 6
ชัยนาท Chai Nat ชน 2,469.75 6
ชัยภูมิ Chaiyaphum ชย 12,778.29 7
ชุมพร Chumphon ชพ 6,010.85 5
เชียงราย Chiang Rai ชร 11,678.37 8
เชียงใหม่ Chiang Mai ชม 20,107.06 9
ตรัง Trang ตง 4,917.52 4
ตราด Trat ตร 2,819.00 4
ตาก Tak ตก 16,406.65 3
นครนายก Nakhon Nayok นย 2,122.00 7
นครปฐม Nakhon Pathom นฐ 2,168.33 6
```

```
//                return o1.getNameThLength() - o2.getNameThLength();
            return Integer.valueOf(o1.getNameThLength()).compareTo(Integer.valueOf(o2.getNameThL
        }
    });

    list.sort(new Comparator<Province>() {
        @Override
        public int compare(Province o1, Province o2) {
            Collator coll = Collator.getInstance(new Locale("th", "TH"));
            if (o1.getNameThLength() == o2.getNameThLength()) {
                return coll.compare(o1.getNameTh(), o2.getNameTh());
            } else {
                return Integer.valueOf(o1.getNameThLength()).compareTo(Integer.valueOf(o2.getNam
            }
        }
    });
```

จนกระทั่ง ผลรวมเท่ากัน

```
"C:\Program ...
ตาก Tak ตก 16,406.65 3
เลย Loei ลย 11,424.61 3
ตรัง Trang ตง 4,917.52 4
ตราด Trat ตร 2,819.00 4
น่าน Nan นน 11,472.07 4
แพร่ Phrae พร 6,538.60 4
ยะลา Yala ยล 4,521.08 4
```

```
public Person(String firstName, String lastName, String nickName, String gender) {
    this.firstName = firstName.trim().substring(0,1).toUpperCase() +
            firstName.trim().substring(1).toLowerCase();
    this.lastName = lastName;
    this.nickName = nickName;
    this.gender = gender;
}

// chain constructor
public Person(String firstName, String nickName) {
    this(firstName, "", nickName, "");
}

/*
    public Person(String firstName, String nickName)
        this.firstName = firstName.trim().substring(
                firstName.trim().substring(1).toLowerCase();
//        this.firstName = firstName;
        this.nickName = nickName;
    }
```

chain constructor

```
"C:\Program ...
peTEr
```

```
 * Created by prasert on 11/8/2014.
 */
public class Person {
    private String firstName, lastName, nickName, gender;

    public Person(String firstName, String lastName, String nickName, String gender) {
//        this.firstName = firstName.trim().substring(0,1).toUpperCase() +
//                firstName.trim().substring(1).toLowerCase();
        setFirstName(firstName);
        this.lastName = lastName;
        this.nickName = nickName;
        this.gender = gender;
    }

    // chain constructor
    public Person(String firstName, String nickName) {
        this(firstName, "", nickName, "");
    }

    public Person(String firstName, String lastName, String gender) {
```

```
"C:\Program ...
Peter

Process finished with exit code 0
```

```
        }
        */
    public Person() {
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
//        this.firstName = firstName;
        this.firstName = firstName.trim().substring(0,1).toUpperCase() +
                firstName.trim().substring(1).toLowerCase();
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
```

```
"C:\Program ...
Peter

Process finished with exit code 0
```

ลูก ไม่มี Constructor เขียน มีแต่ เป็น default ตอน RUN จะใช้ตาม class แม่

```
ConstructorConcept > src > com > prasertcbs > Patient
Main.java | Person.java | Patient.java
```

```
1  package com.prasertcbs;
2
3  /**
4   * Created by prasert on 11/8/2014.
5   */
6  public class Patient extends Person {
7      private float height, weight;
8
9      public float getHeight() {
10         return height;
11     }
12
13     public void setHeight(float height) {
14         this.height = height;
15     }
16
17     public float getWeight() {
18         return weight;
19     }
20 }
```

```
Run Main
"C:\Program ...
Person constructor was called.

Process finished with exit code 0
```

ถูกกำหนดจาก class แม่

```
ConstructorConcept > src > com > prasertcbs > Patient
Main.java | Person.java | Patient.java
```

```
1  package com.prasertcbs;
2
3  /**
4   * Created by prasert on 11/8/2014.
5   */
6  public class Patient extends Person {
7      private float height, weight;
8
9      public Patient() {
10     }
11
12
13     public float getHeight() {
14         return height;
15     }
16
17     public void setHeight(float height) {
18         this.height = height;
19     }
20 }
```

default

```
Run Main
"C:\Program ...
Person constructor was called.
```

```
ConstructorConcept > src > com > prasertcbs > Main
Main.java | Person.java | Patient.java
```

```
1  package com.prasertcbs;
2
3  public class Main {
4
5      public static void main(String[] args) {
6          demo4();
7      }
8
9      public static void demo4() {
10         Patient t = new Patient();
11         Patient t2 = new Patient("Peter", "Parker", "M", "Pete", 170f, 70f);
12         System.out.println(t2.getFirstName() + " " + t2.getLastName() +
13                                 " height = " + t2.getHeight());
14     }
15
16     public static void demo() {
17         Person p1 = new Person();
18         p1.setFirstName("Peter");
19         p1.setLastName("Parker");
20         p1.setGender("M");
```

สมมติว่าใน class มี constructor 3 sout("BB") ตอน RUN demo 4

BB

Peter Parker ...

```
Run Main
Person constructor was called.
Peter Parker height = 170.0

Process finished with exit code 0
```

```java
        setA.add("mango");
        setA.add("banana");
        setA.add("orange");
        setA.add("banana");

        System.out.printf("%s%n", setA);
    }

    public static void demo2() {
        Set<String> setA = new HashSet<String>(Arrays.asList("mango", "banana",
                                                             "orange", "banana"));
        System.out.printf("%s%n", setA);
    }
}
```

```
"C:\Program ...
[banana, orange, mango]

Process finished with exit code 0
```

```java
                                                             "orange", "banana"));
        System.out.printf("%s%n", setA);
    }

    public static void demo3() {
        List<String> fruits = new ArrayList<String>(Arrays.asList("mango", "banana",
                                                                  "orange", "banana"));
        Set<String> setA = new HashSet<String>(fruits);
        Set<String> setB = new LinkedHashSet<String>(fruits);
        Set<String> setC = new TreeSet<String>(fruits);

        System.out.printf("HashSet      : %s%n", setA);
        System.out.printf("LinkedHashSet: %s%n", setB);
        System.out.printf("TreeSet      : %s%n", setC);

    }
```

```
"C:\Program ...
HashSet      : [banana, orange, mango]
LinkedHashSet: [mango, banana, orange]
TreeSet      : [banana, mango, orange]

Process finished with exit code 0
```

```java
public class Int103w05w {

    public static void main(String[] args) {
        //testPerson();
        testGroup();
    }
    static void testPerson() {
        var p0 = new Person("Sampson","Jones");
        var p1 = new Person("Albert","Smith");
        var p2 = new Person("Robert","Dole");
        System.out.println("p0: " + p0);
        System.out.println("p1: " + p1);
        System.out.println("p2: " + p2);
    }
```

เพิ่มมาตัวแรก id = 0

```
Person[id:2,name:Robert,Dole]
Person[id:0,name:Sampson,Jones]
Person[id:1,name:Albert,Smith]}
sorting g: Group{
Person[id:0,name:Sampson,Jones]
Person[id:1,name:Albert,Smith]
Person[id:2,name:Robert,Dole]}
sorting g by firstname: Group{
Person[id:1,name:Albert,Smith]
Person[id:2,name:Robert,Dole]
Person[id:0,name:Sampson,Jones]}
```

Arrays.sort() ปกติ

Arrays.sort(Person.FIRSTNAME);

```java
import java.util.Comparator;

public class Group<T> { // type parameter
    private static final int SIZE = 5;
    private static final int NOTFOUND = -1;
    private T[] box = (T[]) new Object[SIZE];
    private int count;

    public boolean add(T t) {
        if (t == null || locate(t) != NOTFOUND) return false;
        box[count++] = t;
        return true;
    }

    public T find(T t) {
        int i = locate(t);
        return i == NOTFOUND ? null.: box[i];
    }

    private int locate(T t) {
        if (t == null) return NOTFOUND;
        for (int i = 0; i < count; i++) {
            if (box[i].equals(t)) return i;// new t = t return i
        }
        return NOTFOUND;
    }

    public Group sort() {
        if (count < 2) return this;
        Arrays.sort(box, 0, count);
        return this;
    }

    public Group sort(Comparator<T> c) {
        if (count < 2) return this;
        Arrays.sort(box, 0, count, c);
        return this;
    }

    @Override
    public String toString() {
        var b = new StringBuilder();
        b.append("Group{");
        for (int i = 0; i < count; i++) {
            b.append("\n  ").append(box[i]);
        }
        b.append("}");
        return b.toString();
```

```java
lic class Person implements Comparable<Person> {
private static int nextId;
private final int id;
private final Gender sex;
protected String firstname;
protected String lastname;

private static class FirstnameComparator implements Comparator<Person> {
    @Override
    public int compare(Person p0, Person p1) {
        return p0.firstname.compareTo(p1.firstname);
    }
}
private static class LastnameComparator implements Comparator<Person> {
    @Override
    public int compare(Person p0, Person p1) {
        return p0.lastname.compareTo(p1.lastname);
    }
}

public static final Comparator<Person> FIRSTNAME_COMPARATOR = new FirstnameComparator();
public static final Comparator<Person> LASTNAME_COMPARATOR = new LastnameComparator();

public Person(Gender sex, String firstname, String lastname) {
    this.sex = Objects.requireNonNull(sex, "Person's gender cannot be null.");
or      if (sex == null) throw new NullPointerException("Person 's gender cannot be null");
        this.sex = sex;
    this.firstname = Util.isValid(firstname) ? firstname : Util.BLANK;
    this.lastname = Util.isValid(lastname) ? lastname : Util.BLANK;
    this.id = nextId++;
```

```java
        var p2 = new Person(Gender.FEMALE,"Samantha","Albert");
        System.out.println("p0: " + p0);
        System.out.println("p1: " + p1);
        System.out.println("p2: " + p2);
    }
    static void testGroup() {
        var g = new Group<Person>();
        var p0 = new Person(Gender.MALE,"Smith","Johnson");
        var s1 = new Student(Gender.OTHER,"Jame","Simpson",Level.JUNIOR);
        var s2 = new Student(Gender.FEMALE,"Valentine","Rose",Level.FRESHMAN);
        var p3 = new Person(Gender.FEMALE,"Samantha","Albert");
        System.out.println("Add p3: " + g.add(p3));
        System.out.println("Add s2: " + g.add(s2));
        System.out.println("Add p0: " + g.add(p0));
        System.out.println("Add s1: " + g.add(s1));
        System.out.println("Listing g: " + g);
        System.out.println("Sorting g: " + g.sort());
        System.out.println("Sorting g by firstname: " + g.sort(Person.FIRSTNAME_COMPARATOR));
        System.out.println("Sorting g by lastname: " + g.sort(Person.LASTNAME_COMPARATOR)); // sort is order by
    }
}
```

new อันแรก id = 0

add อันแรก สั่งพิมพ์ออกมา จะตามลำดับที่ add

Output - Run (Int103w05t)

```
Listing g: Group{
  Person[id:3,sex:FEMALE,name:Samantha,Albert]
  Student[id:2,sex:FEMALE,name:Valentine,Rose][level:Year One]
  Person[id:0,sex:MALE,name:Smith,Johnson]
  Student[id:1,sex:OTHER,name:Jame,Simpson][level:JUNIOR]}
Sorting g: Group{
  Person[id:0,sex:MALE,name:Smith,Johnson]
  Student[id:1,sex:OTHER,name:Jame,Simpson][level:JUNIOR]
  Student[id:2,sex:FEMALE,name:Valentine,Rose][level:Year One]
  Person[id:3,sex:FEMALE,name:Samantha,Albert]}
Sorting g by firstname: Group{
  Student[id:1,sex:OTHER,name:Jame,Simpson][level:JUNIOR]
  Person[id:3,sex:FEMALE,name:Samantha,Albert]
  Person[id:0,sex:MALE,name:Smith,Johnson]
  Student[id:2,sex:FEMALE,name:Valentine,Rose][level:Year One]}
Sorting g by lastname: Group{
  Person[id:3,sex:FEMALE,name:Samantha,Albert]
  Person[id:0,sex:MALE,name:Smith,Johnson]
  Student[id:2,sex:FEMALE,name:Valentine,Rose][level:Year One]
  Student[id:1,sex:OTHER,name:Jame,Simpson][level:JUNIOR]}
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```

Test Enum

```java
    private static void testEnum() {
        //print enum can be use 2 versoin
        for (Object value : Gender.values()) {
            System.out.println("Gender: " + value);
        }
        for (var value : Level.values()) {
            System.out.println("Level: " + value);
        }


        for (var value : Level.values()) {
            System.out.println("Level: " + value + ", Year: " + value.getYear());
        }
        var lvl = Level.valueOf("FRESHMAN");
        System.out.println("Level.FRESHMAN: " + lvl.getYear() + ", ordinal " + lvl.ordinal());
    }
}
```

```java
public enum Level {
    FRESHMAN(1),      // ordinary = 0
    SOPHOMORE(2),     // ordinary = 1
    JUNIOR(3),
    SENIOR(4);
    private final int year;
    private Level(int year){ this.year = year; }
    public int getYear(){ return year ;}
```

ค่าหลังข้างใน (lvl.ordinal())

FRESHMAN เหมือน subclass ของ Level อีกที และเป็น Object ไปในตัว

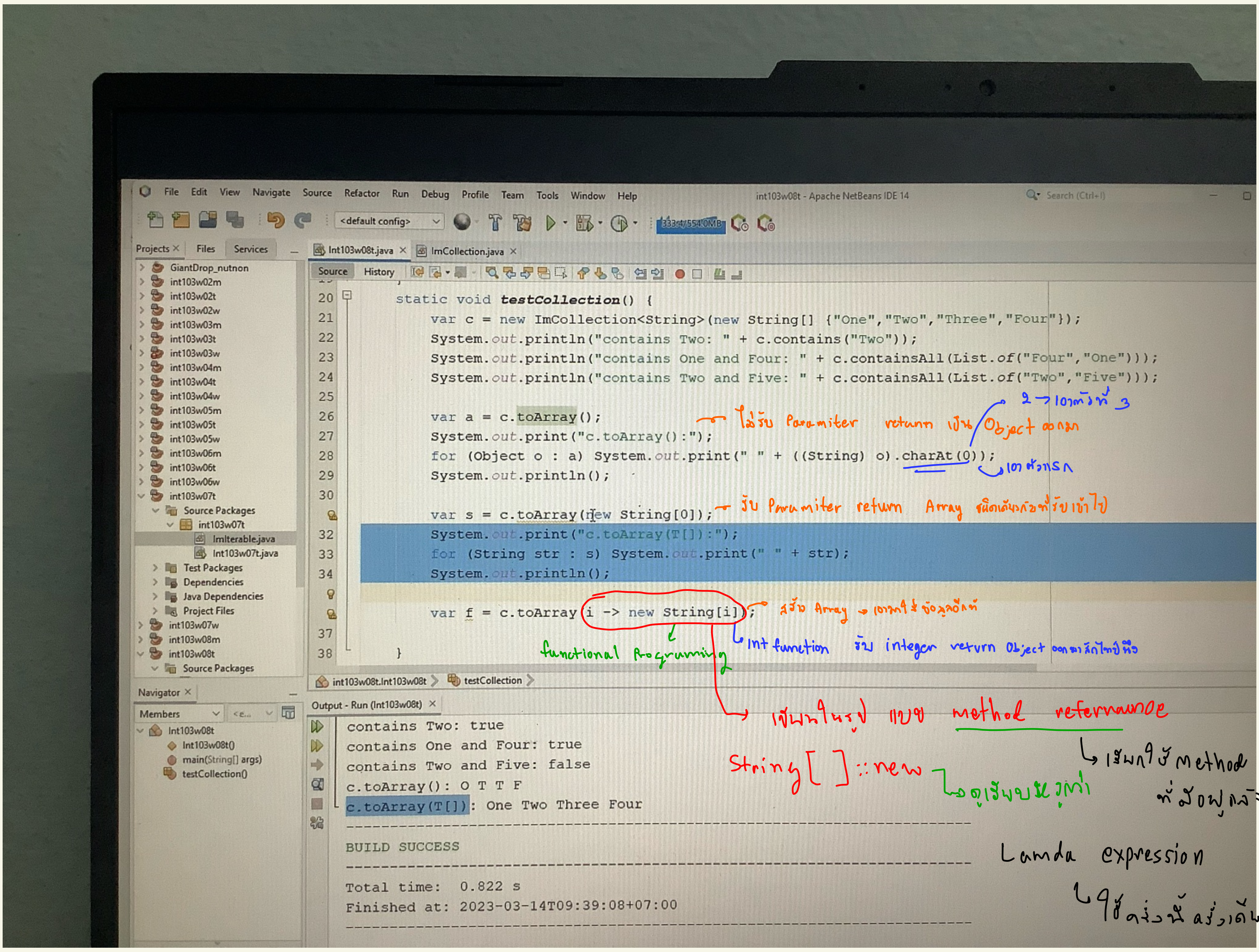
Output - Run (Int103w05t)

```
------------------------< int103w05t:int103w05t >------------------------
Building int103w05t 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.1.0:exec (default-cli) @ int103w05t ---
Gender: MALE
Gender: FEMALE
Gender: OTHER
Level: FRESHMAN
Level: SOPHOMORE
Level: JUNIOR
Level: SENIOR
Level: FRESHMAN, Year: 1
Level: SOPHOMORE, Year: 2
Level: JUNIOR, Year: 3
Level: SENIOR, Year: 4
Level.FRESHMAN: 1, ordinal 0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.678 s
Finished at: 2023-04-01T18:14:59+07:00
------------------------------------------------------------------------
```

Enum สร้าง Object เพิ่มไม่ได้

ถ้าจะสร้าง ต้องไปแก้ที่ ไฟล์ / เปลี่ยนต้น

แปลงเป็น Arrays



```
20    static void testCollection() {
21        var c = new ImCollection<String>(new String[] {"One","Two","Three","Four"});
22        System.out.println("contains Two: " + c.contains("Two"));
23        System.out.println("contains One and Four: " + c.containsAll(List.of("Four","One")));
24        System.out.println("contains Two and Five: " + c.containsAll(List.of("Two","Five")));
25
26        var a = c.toArray();
27        System.out.print("c.toArray():");
28        for (Object o : a) System.out.print(" " + ((String) o).charAt(0));
29        System.out.println();
30
          var s = c.toArray(new String[0]);
32        System.out.print("c.toArray(T[]):");
33        for (String str : s) System.out.print(" " + str);
34        System.out.println();

          var f = c.toArray(i -> new String[i]);
37
38    }
```

```
contains Two: true
contains One and Four: true
contains Two and Five: false
c.toArray(): O T T F
c.toArray(T[]): One Two Three Four
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.822 s
Finished at: 2023-03-14T09:39:08+07:00
------------------------------------------------------------------------
```


Stream API → Iterator
- Internal Iterator stream → เราส่ง function ไปให้ได้อย่างเดียว → control ไม่ได้, มั่นคง ดวงดี
- External Iterator → ใช้ร่วมกับ อยู่ข้างนอก (Iterator in Java) → ควบคุมได้
  - วนLoop เอง → เราต้องจำข้อมูลเส้นทางได้

process

ข้อมูลเยอะ มาก → น้อยใช้ ชัวร์?
- .parallelStream() → มี cpu หลายตัว แตกออก มาหลาย multi verse
- or .parallel() → แตกตัวไปช่วยกัน RUN

.filter(st → st.contain("e")) — one Three
.map(String :: length) → auto boxing — ได้ 3, 5
- เปลี่ยนจาก string ของสิ่งหนึ่ง ไปเป็น string ของอีกอย่าง
- Return เป็น Object

.map(s → s.charAt(2)); ได้ e, r

- .Parallel()
- .Sequential() → ได้ 4 3 5 4

หลักการ Stream

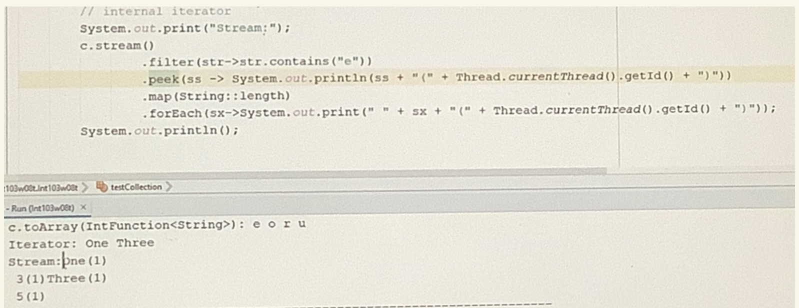
1. create stream
2. process stream → หลายครั้ง .filter(), .parallel(), .map()
3. ปิดท้าย Stream → for each เอาไว้ ปิดท้าย Stream → ทำได้ครั้งเดียว

- .sequential() → เรียงข้อมูลตามที่ใส่
- .skip() → ข้ามไปทำงานที่ใส่, .Limit() → บอกลิมิต, .distinct() → ซ้ำก็ตัดออก
- .sorted() → เรียงค่าลำดับ
- .peek() → เอาไว้สำหรับการ Debug

- .flatMap() → มี Stream ใน stream ต้องการให้มันแตกออก
- mapToInt → map จาก Object ให้ไปเป็น primitive
- .boxed() → boxzing → ทำให้กลับมาเป็น Object
- .takeWhile / dropWhile() → คล้าย while Loop หลุด Loop เมื่อผิด
  - ถ้าในแต่เป็น Stream แล้ว T จะไม่ทำ ตอนปิดท้ายF

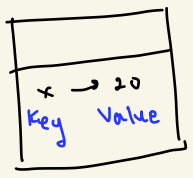
c .allMatch(consumer) → ทุกตัวต้อง กำหนดไว้ใน consumer ต้องตรงเงื่อนไข → True
anyMatch(consumer) → บางตัวก็ได้ → True ไม่ตรงเลย False
noneMatch(consumer) → ต้องไม่มีสักตัว ตรงเงื่อนไข → True

reduce



```
// internal iterator
System.out.print("Stream:");
c.stream()
        .filter(str->str.contains("e"))
        .peek(ss -> System.out.println(ss + "(" + Thread.currentThread().getId() + ")"))
        .map(String::length)
        .forEach(sx->System.out.print(" " + sx + "(" + Thread.currentThread().getId() + ")"));
System.out.println();
```

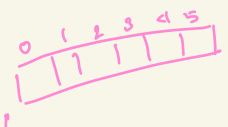
```
c.toArray(IntFunction<String>): e o r u
Iterator: One Three
Stream:One(1)
 3(1)Three(1)
 5(1)
```

Hashtable → 

ค่าพารามาก → hash fuction ← รู้ X ← พลิพพ์ 20 → x ,20 จับคู่กัน

ลักษณะ: เหมือนชื่อด้วย map (1 ค่า จับกับ 1 ค่า)

map
- Hash map → ตารางชั้น
- map
- Dictionary
- Associative Array → แบ่งเป็นช่องๆ เหมือนมานั่งจับคู่กัน ไม่ต้องเป็นตัวเลข เป็นอะไรก็ได้ ที่ไม่เหมือนกัน



ความต่าง Array List กับ Vector

สร้าง Object ขึ้นมา แล้ว จับคู่กันใน map

Treeset → ข้างใน มี map → หน้าบ้านเป็น set แล้ว ข้างในเป็น Tree map

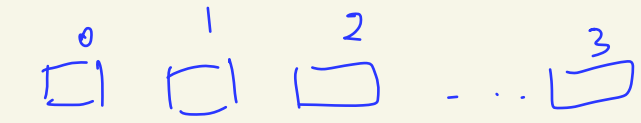
```
Set<String> x = new TreeSet<>();
```

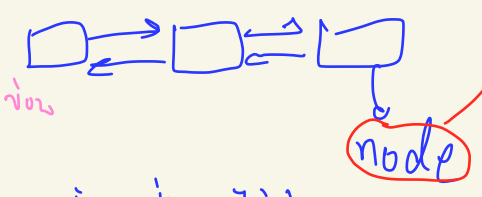
```
List<String> li = new LinkedList<>();
List<String> al = new ArrayList<>();
```

List → กลุ่มก้อนข้อมูลที่มีลำดับ มีลำดับ ไม่ซ้ำกันเอง

ของ ที่เป็น List มี Array อยู่ข้างใน

implement ด้วย LinkList → แกรฟอันมีลำดับไป อันดับไป

List 

LinkedList 

ความเร็วข้อมูล เทียบข้อมูลออกเร็ว → ใช้เวลาเท่ากันหมด

Delete Insert ข้อมูล

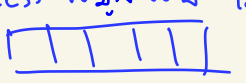
data stucter

```
transient Node<E> first;
transient Node<E> last;
```

ไม่ตาย

↳ nested class

↓ ตอน Add ข้อมูล จะสร้าง node อันใหม่ขึ้นมา แล้วเชื่อมกับ List

ArrayList  Access ข้อมูลข้างใน ไม่ดีเลย Insert delete ไม่ดีตามพื้นฐานโครงสร้าง

↳ ก้อน Array ไว้ข้างหลัง → เข้าถึงได้เร็ว

→ ลบ 5 ข้างหน้า ต่างกัน

→ ทุกอย่าง ตัวเลขต้องพังพู่ ข้างหลังขยับไปเรื่อยๆ

Treeset → Focus data stucter

Hash set → เก็บ ลำดับ

Enum Hash set → set ที่ข้อมูล ชนิด Enum

No duplicate key

TreeSet/Map Vs HashSet/Map

skipList → กระโดดไปไกลๆได้

Iterator → Remove ก่อนได้

method ของ Array

```
System.arrcopy(this,
```

this → array อยู่แล้ว 2 ตัว

array ตัวเก่า เป็น Object อยู่แล้ว

↳ เอาก็คือ Array ตัวเก่า ลบ ข้อมูล ออกตัวใหม่

ตัวใหม่ : `Arrays.copyOf(array, array.length)` → คือ Array ตัวใหม่ แล้ว Copy ข้อมูลมาจาก Array ตัวใหม่

↳ ยังไม่มี Array มีแต่ ตัวแปร ชี้ Array

ตัวเก่า

ความยาว

Iterator → รอรับตามลบ ข้อมูลออก

innitualrized ค่า ของ array new + Type , ต้องใส่ข้อมูล , ไม่จำเป็นต้องบอกว่าเก็บข้อมูลได้กี่ตัว

```
Iterator <String> it = new ImIterable <>( new String[] {"one", "two", "Three"} );

var i = it.Iterator();

while (i.hasNext()) {
    System.out.println("While " + i.next());
}
```

Basic

fuctinon ที่อยู่แล้ว เอามาใช้เลย method Ref

เริ่มจากต้น ถึง ปลายเลย Iterator

```
.foreach ( s → System.out.println(" " + s) )
```

รับ consumer

เริ่มตรงจุดไหนก็ได้ Iterable consumer

```
.forEachRemaining (System.out::println);
```

Collection → Add ข้อมูลได้ , ต้อง implement ทำsometing → Ex ลบ , เพิ่ม แล้วแต่เราทำ

คล้าย

set {
- addAll (collection) → union
- removeAll (collection) → different (-) → เอา collection มา ลบกัน
- retainAll (collection) → intersec
  ↳ เอาอันที่ซ้ำกัน

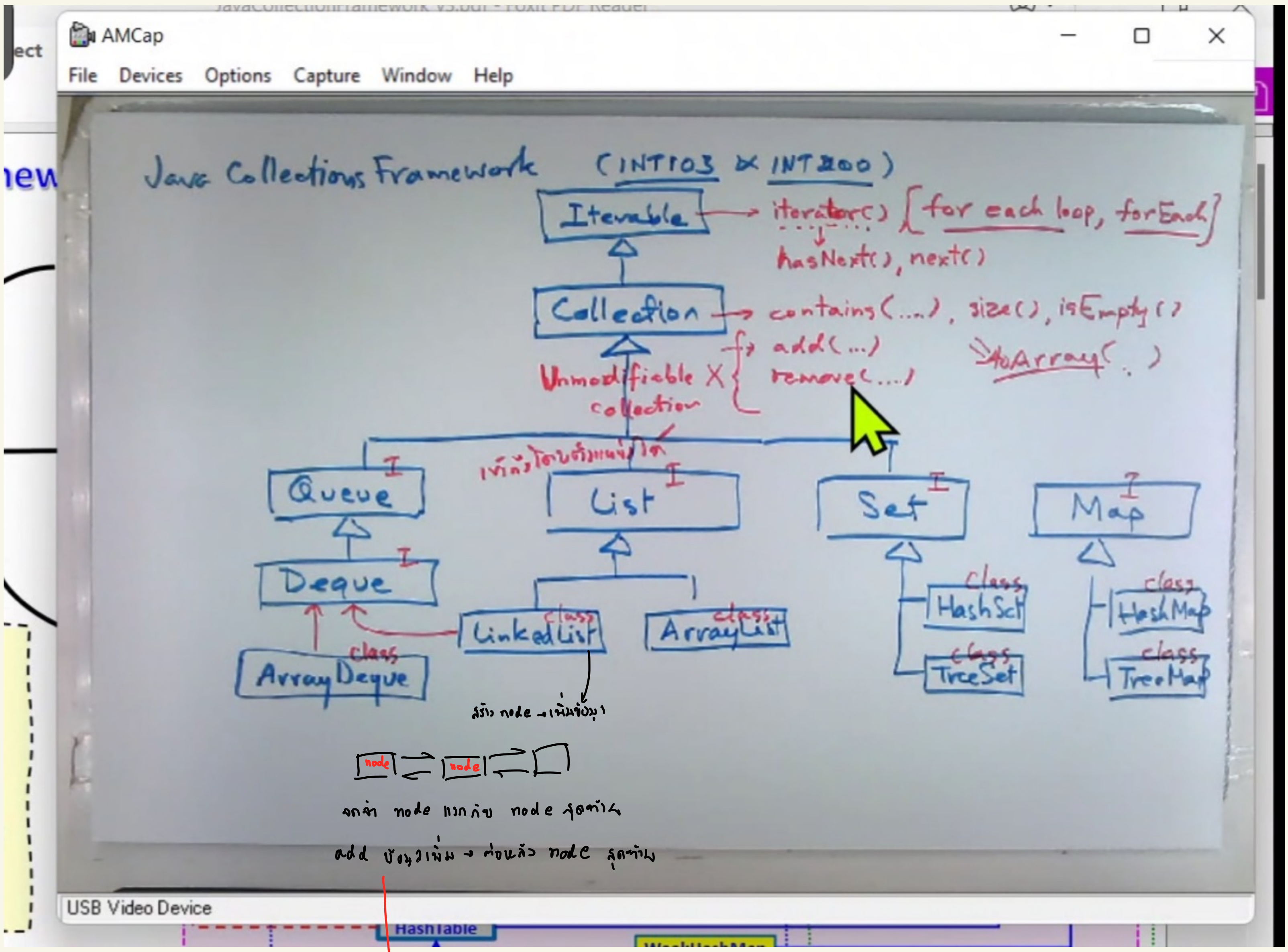
offerce = add(e) → เอาข้อมูลใส่

peek() = element() → อ่านค่าที่หัวแต่ไม่เอาออก

poll() = remove() → เอาที่อยู่หัวทิ้งไป

Return Special Value — แต่: throw exception

↳ Ex หาไม่เจอ return -1



↳ เพิ่ม / ลบ ที่อยู่ข้างใน มีประสิทธิภาพ

Array List ต้อง ขยับข้อมูล

ทำหน้าที่เป็น queue / stack

ensureCapasity (int minCapasity)

Array List ✱ — link List ไม่มี

↳ รู้อยู่แล้วว่าจะ add เพิ่มกี่ตัว → ขยายตามที่เพิ่ม

trimToSize() → ArrayList ✱

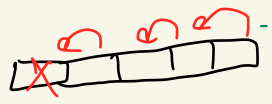
↳ ย้ายบ้านไปอยู่บ้านที่พอดี

map → put (key, value) (เอาข้อมูลใส่ไป)

→ get (key) → หาข้อมูล

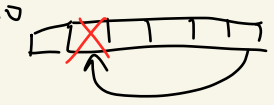
- entrySet() → มี map อยู่ ตามการอ่านเป็น set
  ↳ return เป็นเซ็ต set ของ Entry
- keySet() → return เป็น key

Remove → แล้วเราจะลบยังไง



```
pubile void replace (int i) {
    --count;
  for (int x = i; x < count; x++) {
    array [x] = array [x+1];
  }
array [ count ] = null;
}
```

Remove → แล้วเราเอาตัวสุดท้ายมาต่อ



```
pubile void replace ( ) {
array [i] = array [--count];

array [caunt ] = null
}
```

Iterable → ฝังมาพร้อม Array List / หรือ List

```
pubilc class ExampleList <T> implement Iterable <T> {
private List <T> myList = new ArrayList <>( );
```

เพิ่มค่าเข้า myList

```
pubile void add (T t) {
    myList.add (value);
}
```

```
@ Override
pubilc Iterator <T> iterator ( ) {
  return new MyIterator <T> (myList);
}
```

Iterable มี fucntion Iterator ซึ่ง Iterator ใช้ Default ถ้าไม่สร้าง จะใช้ Default

ของ type ด้วย

ส่ง myList เข้าไปใช้ใน Iterator

```
pubilc class MyIterator <E> implement Iterator <E> {
  int currentPosition; → สร้าง Index ดูว่า Iterator ของเราดึงตัวที่เท่าไหร่แล้ว
  List <E> list; → สร้าง List เพื่อเก็บว่า ตัว List เป็นตัวที่เท่าไหร่
  pubile MyIterator (List <E> lists) {
    this.lists = lists;
  }
```

สร้าง Constructor รับค่า list เข้ามา ใช้ใน hash next กับ next

```
  @Override → ดัง currentPosition > return false ออกไป
  pubilc boolean hash next ( ) {
    if (lists.size( ) >= currentPosition + 1) {
      return true;
    }
    return false
  }
  @Override
  pubilc E next ( ) {
```

ไล่ List ลำดับต่อไป

```
    E value = lists.get(curren Position);
    currentPosition ++;
    return value;
  }
}
```

Iterable → เรียกใช้งาน

```
ExampleList<String> exampleList = new ExampleList<>();
```

```
exampleList.add("Test 1"); → 0
exampleList.add("Test 2"); → 1
exampleList.add("Test 3"); → 2
exampleList.add("Test 4"); → 3
exampleList.add("Test 5"); → 4
exampleList.add("Test 6"); → 5
exampleList.add("Test 7"); → 6
```

ตัวแปรเก็บค่าอะไร กำหนดตัวแปร

```
for ( String str : exampleList) {
    System.out.println (str);
}
```

output

Test 1
Test 2
Test 3
Test 4

output

Test 1
Test 3
Test 5
Test 7

```
@Override
public E next() {
    E value = lists.get(currentPosition);
    currentPosition += 2;
    return value;
}
```

Compareble → บอกว่า สามารถเปรียบเทียบได้

comperator → ตัวเปรียบเทียบ

```
class Item {

   private string name;
   private string category;
   private float price;
   private int stocks;

 pubilc Item ( String name; String category, float price, int stocks ) {
      this.name = name;
      this.catagory = catagory;
      this.price = price;
      this.stocks = stocks
 }
```

สร้าง getter, setter ออกมา

```
List<Item> items = new ArrayList<>();
items.add (new Item ( "Water", "Drinks", 10F, 100));
items.add (new Item ( "Lays", "Snacks", 22.5F, 20));
items.add (new Item ( "Chocolate", "Snacks", 55.55F, 50));
items.add (new Item ( "Pencil", "Tools", 13.25F, 1000));
items.add (new Item ( "paper", "Tools", 2.5F, 10000));
items.add (new Item ( "Coke", "Drinks", 18F, 20));
```

สร้าง Comparator → เรียงลำดับ

ใช้ Comparaton

```
items.sort (new Comparator<Item>() {
  @Override
   public int compare ( Item o1, Item o2 ){
     return o1.getName.compareTo (o2.getname());
   }
});
```

ขึ้นอยู่กับว่า เราเรียงลำดับ อะไร Ex จะทำให้เรียงตามชื่อ (Name)

สนใจตัวใหญ่ เล็ก

ถ้า compareToIgnoreCase → จะไม่สน

→ เรียงตามตัวอักษร น้อย ไป มาก

Output
C
:
P
:
W

ค่า มาก ไป น้อย เติม (-) ข้างหน้า

Loop เพื่อแสดงค่าออก

```
for ( Item item : items ) {
   System.out.printf (
     "name: %s, Catagory: %s, Price: %.2f, Stock : %d%n",
       item.getName(), item.getCategory(), item.getPrice(), item.getStocks()
   );
```

เรียงตามอะไร ?? → กรณี comparator

↳ Comparator → เรียง 1 ประเภท

→ ถ้าเรียง String ใช้ compare To

• ตามราคา

```
@Override public int compare (Item o1, Item o2) {
    return (int) (o1.getPrice() - o2.getPrice());
}
```

→ น้อยไปมาก

มากไปน้อย ใส่ - ไว้หน้า ( )

↳ Stocks ก็ได้

กรณี comparable

```
class Item implement Comparable <T> {
   ⋮
```

→ Implement mathod compareTo

ข้อดี : เรียง มากกว่า = 2 ประเภท

Ex เรียงตาม Catagory ก่อน แล้วค่อยเรียงตามชื่อ

```
@Override
public int compareTo(Item o) {
 if ( this.getCategory().equalsIgnoreCase(o.getCategory()) ){
   return this.getName().compareTo(o.getName());
 }
 return this.getCategory().compareTo(o.getCategory());
}
```

→ เปรียบเทียบ Object

เช็คว่า Catagory เท่ากันมั้ย

↳ ถ้าเท่ากันก็ เรียง name

→ เรียกใช้

```
// items.sort((Comparator <? super Item >) items);
```

เอาไว้ไหม ไม่รู้ AOV

```
Collections.sort(items)
```

ของต่าคู

```
if ( !(this.getCategory().equalsIgnoreCase(o.getCategory())) ){
   return this.getCategory().compareTo(o.getCategory());
}
return this.getName().compareTo(o.getName());
```

↳ Catagory ไม่เท่ากัน เรียง ก่อน ค่อยเรียง Nam

Map

```
HashMap <String , Integer>   maps = new HashMap<>();
```
Key Value

```
maps.put("A", 10);
maps.put("B", 20);

for (Map.Entry<String, Integer> entry : map.entrySet()) {
    String key = entry.getKey();
    int value = entry.getValue();
}

System.out.println("key: " + key);
System.out.println("key: " + value);
```

printทุก ตอนวน เรียงมั่ว

ดู comparator ด้วยได้

```
TreeMap<String, Integer> maps = new TreeMap<>();
```
Key = String, Value = Integer

```
maps.put("A", 10);
maps.put("B", 20);

for (Map.Entry<String, Integer> entry : map.entrySet()) {
    String key = entry.getKey();
    int value = entry.getValue();
}

System.out.println("key: " + key);
System.out.println("key: " + value);
```

ใช้งาน
ตาม ค่ากับ
key
A
B

```
pubile static String gradecal(int score) {

    Treemap<Integer, String> grads = new TreeMap<>();

    grads.put(50, "D");
    grads.put(0, "F");
    grads.put(80, "A");
    grads.put(60, "C");
    grads.put(70, "B");

    return grads.floorEntry(score).getValue;
}
```

floorEntry(score) → ส่งค่า key → ค้นดู score = 55 ก็ดูออก D
เพราะ เป็น Floor

getValue → เพื่อส่ง Value ออกมา

Stack

Stack <String> stacks = new Stack<>();

		index
stacks.push("coconut");	- 1	0
stacks.push("apples");	- 2	1
stacks.push("orangs");	- 3	2
stack push("melon");	- 4	3

System.out.println(stacks); → output

เรียงตามที่เรา add

[coconut, apple, orage, melon]

system.out.println(stacks.peek()); → output melon

↳ เอาตัวสุดท้ายมาดู ไม่ได้ลบ

System.out.println(stacks.pop()); → output melon

↳ เอาตัวสุดท้ายออกมาดู และ ลบ ออก

System.out.println(stacks.search("apple")); → output 2

ถ้าใส่อะไรมั่วๆ ที่ไม่ได้อยู่ใน stack

จะเป็น -1

System.out.println(stacks.get(1)); → Output apple

System.out.println(stacks.contains("apple")); → output true → มี
(boolean)

false คือไม่เจอ / ไม่มี

System.out.println(stacks.empty()); → output fasle
(boolean)

↳ เช็คว่า stack ว่างเปล่า entry มั้ย

Queue

```
Queue <String> queues = new LinkedList<> ();
queues.offer("1st");
queues.offer("2nd");
queues.offer("3th");
queues.offer("4th");
```

```
System.out.println(queues);
```

output [1st, 2nd, 3th, 4th]

เรียงตามลำดับ add เข้า

```
System.out.println(queues.peek());
```

↳ output 1st → เอาตัวแรก → ไม่ลบ

```
System.out.println(queues.poll());
```

output 1st → เอาตัวแรกสุด ลบ

```
System.out.println(queues.remove("3th"));
```

ลบได้ output true