Practices on Java Programming:

extends/implements Interfaces, extends Classes, Generic Interfaces/Classes/Methods,

Nested Classes, Inner Classes, Anonymous Classes, Local Classes, Iterator<T>, Iterable<T>, Comparator<T>, Comparable<T>

Bag<T>, Sequence<T>, and Condition<T>

INT105 Computer Programming II @ 2018/2

School of Information Technology (SIT)

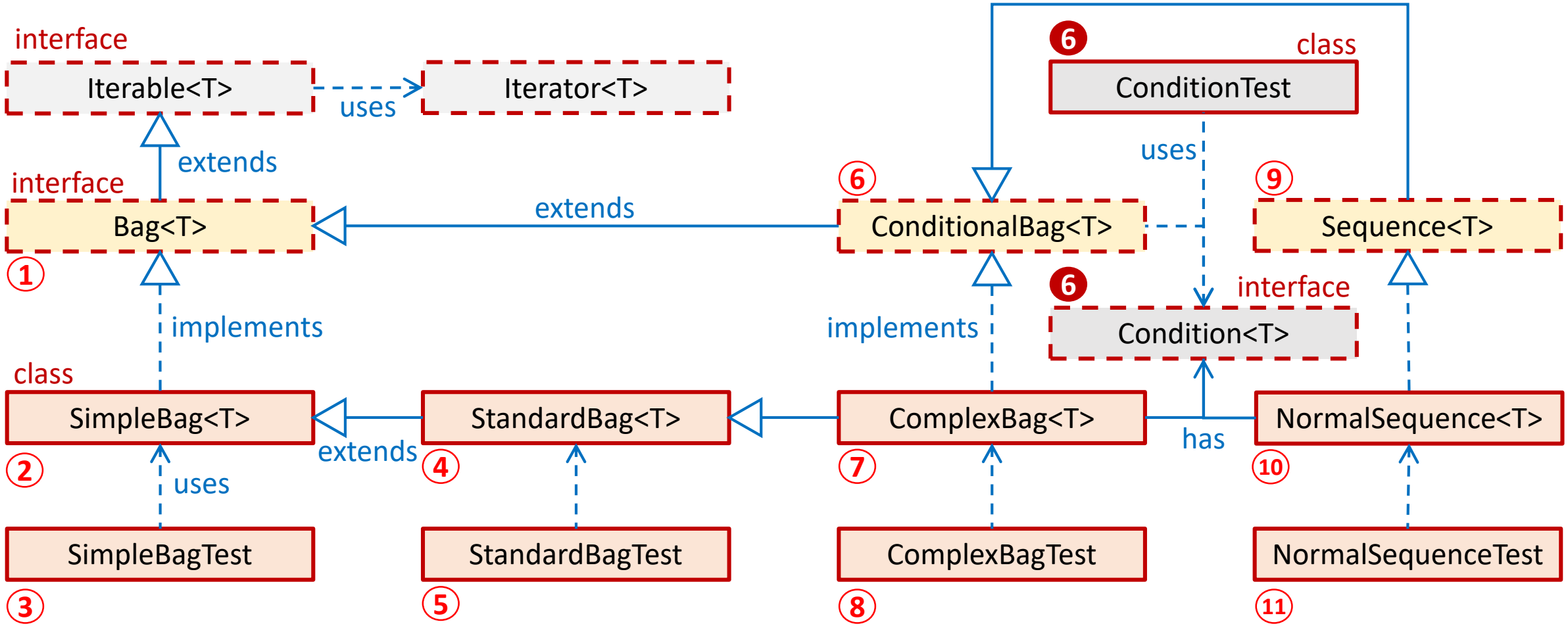
King Mongkut's University of Technology Thonburi (KMUTT)

V.1

Bag<T>, Condition<T>, and Sequence<T>

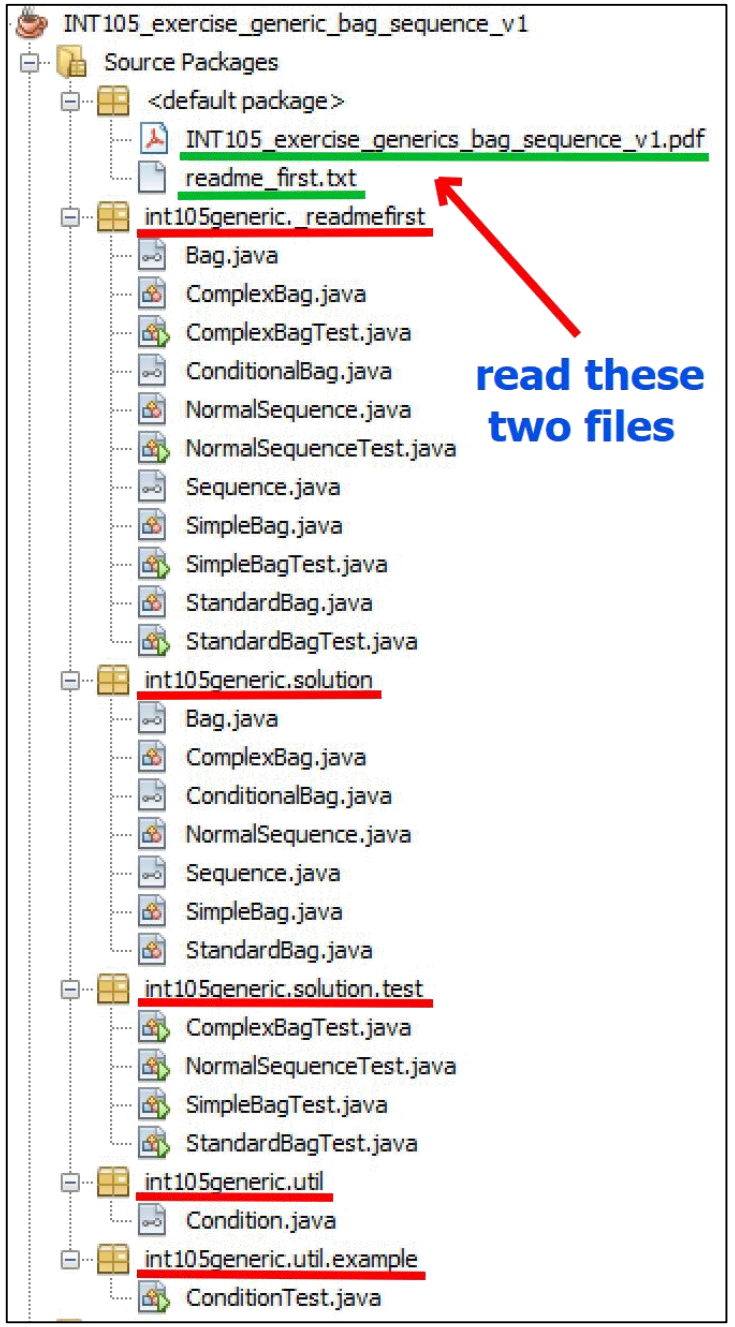
Problems 1 – 11:

Generics, Interfaces, Classes, Nested Classes



โจทย์แบบฝึกหัดพร้อมเฉลยเรื่อง generics, interfaces, classes, nested classes, iterator, comparator

- ให้ฝึกหัดทำโจทย์ตามลำดับหมายเลขข้อที่เขียนในวงกลมสีแดง (จากรูปในหน้าที่ 2) เนื่องจาก แต่ละข้อสัมพันธ์ต่อเนื่องกันจากข้อก่อนหน้า
- อย่างแรกที่ควรทำคือ อ่านโจทย์ใน **readme_first.txt** และ **pdf file** ที่อยู่ใน src folder ของ netbeans project zip file ที่ให้ไว้ใน Classroom on Demand
- มีบางเรื่องที่ น.ศ. ควรเรียนรู้ด้วยตนเองเพิ่มเติมจากเฉลยที่มีให้ เช่น เรื่อง variable arguments (ใน Condition<T> andAll() และ orAll() methods), default/static methods ของ interfaces, การ extends interface, การใช้ generic methods, การใช้ generic ซ้อน generic, fail-fast/fail-safe iterator, การใช้ static/non-static block
- Folder: **int105generic._readmefirst** เก็บโจทย์
- Folder: **int105generic.solution** และ **int105generic.solution.test** เก็บเฉลย
- Folder: **int105generic.util** และ **int105generic.util.example** เก็บ **Condition<T> interface** และตัวอย่างการใช้ method ต่าง ๆ ของ Condition<T>



Bag<T>, SimpleBag<T>, SimpleBagTest

- **Bag<T>** คือ การกำหนดลักษณะของถุงใส่สิ่งของ (คือเป็น interface) โดยสิ่งของที่ใส่ ต้องเป็นประเภทเดียวกัน (คือเป็น generic type T) ไม่มีระเบียบ (คือ สลับลำดับสิ่งของไปมาได้) มีของซ้ำได้ (คือ ใส่สิ่งของลงในถุงโดยไม่ตรวจสอบว่ามีสิ่งของชิ้นนั้นอยู่ในถุงแล้ว) ใส่สิ่งของลงในถุงเพิ่มได้ (คือมี **add(T t) method**) เอาสิ่งของออกจากถุงได้ (คือมี **remove(T t) method**) ค้นหาว่ามีสิ่งของชิ้นใดอยู่ในถุงนั้นหรือเปล่าได้ (คือมี **contains(T t) method**) และไล่ดูสิ่งของแต่ละชิ้นในถุงได้ (คือ เป็น interface ที่ extend **Iterable<T>**)
- **SimpleBag<T>** เป็นถุงที่ใส่สิ่งของได้จริง ๆ (คือเป็น concrete class ที่ implement Bag<T>) ขณะที่ไล่นับดูสิ่งของนั้น สามารถเอาสิ่งของออกได้ด้วย (คือมี **iterator** ที่ implement **remove() method**), นอกจากนี้แล้ว ถ้าหากว่า ขณะที่กำลังไล่ดูสิ่งของในถุงนั้น มีการขัดจังหวะด้วยการใส่สิ่งของเพิ่มหรือเอาสิ่งของออกจากถุง (โดยไม่ได้ทำผ่าน iterator ตัวนั้น) จะทำให้การไล่ดูสิ่งของนั้นผิดพลาดได้ (คือ iterator ตัวนั้นทำงานผิดพลาดได้ หากมีการ add หรือ remove สิ่งของ ระหว่างที่ iterator ตัวนั้นทำงานอยู่ – **fail unpredictably**)
- **SimpleBagTest** เป็นการทดสอบการใช้ SimpleBag<T>

StandardBag<T>, StandardBagTest and fail-fast iterators vs. fail-safe iterators

- **StandardBag<T>** ทำได้ทุกอย่างเหมือน SimpleBag<T> (คือ เป็น class ที่ extend SimpleBag<T>) แต่ StandardBag<T> จะยกเลิกการไล่ดูสิ่งของในถุง หากมีการขัดจังหวะด้วยการใส่สิ่งของเพิ่มหรือเอาออกจากถุงผ่านช่องทางอื่น ขณะที่กำลังไล่ดูสิ่งของอยู่ (คือ iterator ที่มีให้ เป็น **fail-fast iterator**)
  - ในทางทฤษฎี มี Iterator อีกแบบหนึ่ง เรียกว่า **fail-safe iterator** หมายถึง iterator ที่ไม่ยอม fail (คือ safe from fail หรือ free from fail หรือปลอดภัยจากการ fail) คือ iterator ที่เสมือนว่าใช้วิธีถ่ายรูปเก็บไว้ แล้วไปไล่ดูในรูปแทน (หมายถึง สร้าง copy ของ collection นั้นไว้ แล้ว iterate บน copy แทน) ซึ่งผลเสียก็คือ เสมือนเป็นการ iterate อยู่บนชุดข้อมูลเก่า ซึ่งไม่เป็นปัจจุบันแล้ว
  - ใน collection ต่าง ๆ ที่ java มี iterator มาให้นั้น ส่วนใหญ่ (ถ้าไม่ใช่ทั้งหมด) เป็น fail-fast iterator
- **StandardBagTest** เป็นการทดสอบการใช้งาน StandardBag<T>

ConditionalBag<T>, ComplexBag<T>, ComplexBagTest

- **ConditionalBag<T>** เป็นการกำหนดลักษณะถุงใส่สิ่งของ เพิ่มเติมจาก Bag (คือ interface ที่ extends Bag<T> interface) คือสามารถค้นหาสิ่งของในถุงหรือเอาสิ่งของออกจากถุงโดยบอกลักษณะของสิ่งของที่ต้องการค้นหาหรือต้องการเอาออกได้ โดยสามารถทำได้ทั้งแบบค้นหาสิ่งของหรือเอาสิ่งของออกทีละรายการที่ตรงตามลักษณะที่ต้องการ (คือ ConditionalBag<T> มี **contains(Condition<T> c) method** และ **remove(Condition<T> c) method**) หรือแบบค้นหาสิ่งของหรือเอาสิ่งของออกทุกรายการที่ตรงตามลักษณะที่ต้องการ (คือ ConditionalBag<T> มี **allContains(Condition<T> c) method** และ **removeAll(Condition<T> c) method**) นอกจากนี้แล้ว ยังมีการไล่ดูสิ่งของได้สองรูปแบบ คือ แบบที่ Bag มีตามปกติ (คือมี **iterator() method**) และแบบที่สามารถกำหนดลักษณะของสิ่งของที่ต้องการไล่ดูได้ (คือมี **iterator(Condition<T> c) method**)
- **ComplexBag<T>** เป็น ConditionalBag<T> ที่ใส่สิ่งของได้จริง ๆ (คือเป็น concrete class ที่ implements ConditionalBag<T>) และยังเป็น Bag<T> ที่สามารถกำหนดลักษณะของสิ่งของที่จะสามารถใส่ลงไปในถุงได้ด้วย (คือ มี **constructor** ที่ใส่ **condition** เข้าไปในถุงได้ โดย condition ที่ใส่เข้าไปในถุงนี้ ใช้สำหรับตรวจสอบว่าเฉพาะสิ่งของที่ผ่านการตรวจสอบด้วย condition นี้เท่านั้น ที่จะสามารถใส่เข้าไปในถุงได้)
- **ComplexBagTest** เป็นการทดสอบการใช้งาน ComplexBag<T>

Sequence<T>, NormalSequence<T>, NormalSequenceTest

- **Sequence<T>** เป็นการกำหนดลักษณะของ ConditionalBag<T> ที่รักษาลำดับของสิ่งของที่ใส่เข้ามาในถุงได้ (คือเป็น interface ที่ extends ConditionalBag<T>) และสามารถจัดเรียงลำดับของสิ่งของในถุงตามธรรมชาติของสิ่งของนั้นได้ (คือ Sequence<T> มี **sort() method** และ T ต้อง implements Comparable<T> จึงจะสามารถมีการจัดลำดับแบบธรรมชาติได้) หรือกำหนดวิธีการเรียงลำดับเป็นแบบอื่นก็ได้ (คือ Sequence<T> มี **sort(Comparator<T> c) method**)
- **NormalSequence<T>** เป็น Sequence<T> ที่ใส่สิ่งของได้จริง ๆ (คือเป็น concrete class ที่ implements Sequence<T>)
- **NormalSequenceTest** เป็นการทดสอบการใช้งาน NormalSequence<T>