Note Searching & Sorting

Lab #3

COMP3021 2022 Spring

ChengPeng Wang(cwangch@cse.ust.hk)
Yiyuan Guo(yguoaz@cse.ust.hk)
Bowen Zhang(bzhangbr@cse.ust.hk)
Heqing Huang(hhuangaz@cse.ust.hk)

- **Objectives of this lab**

Learn How to Make Classes Comparable

Implements the interface `Comparable` and override function `compareTo()`

Learn How to Sort Objects Using Build-In Functions

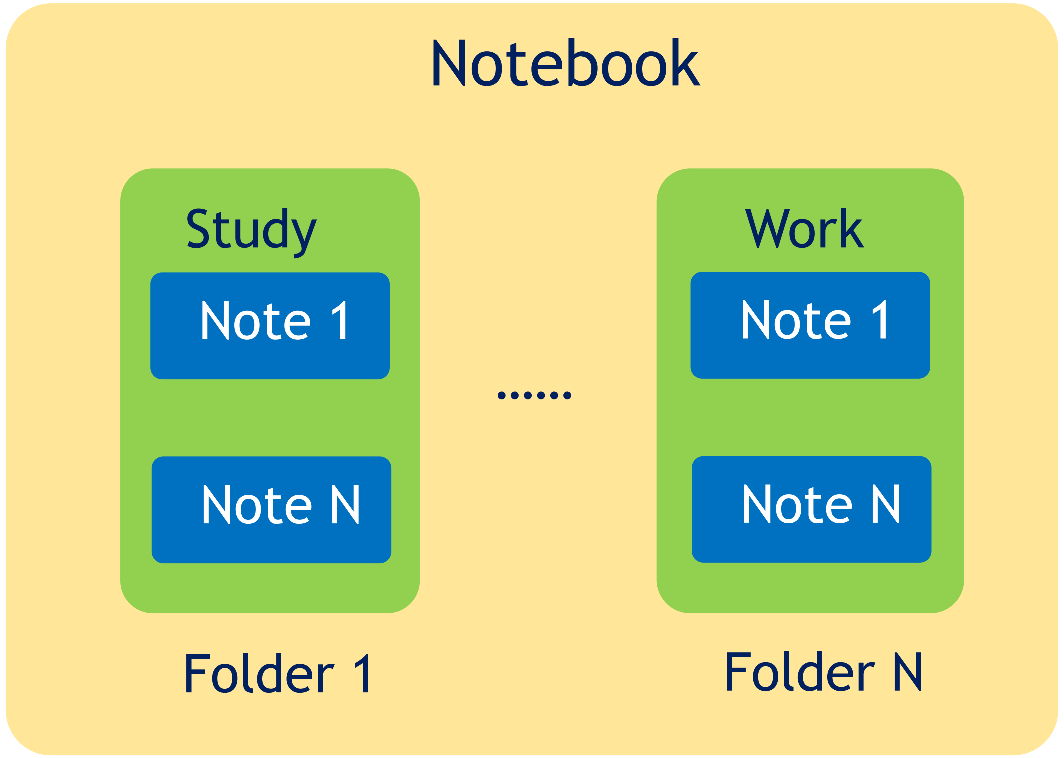
`Collections.sort(List<Comparable Objects>)`

Practice Build-In Methods of String Class

`String.split(), String.toLowerCase(), String.contains(), …`

- **Overview**

This lab session is based on your code implemented in Lab2, please pull your codes first ☺

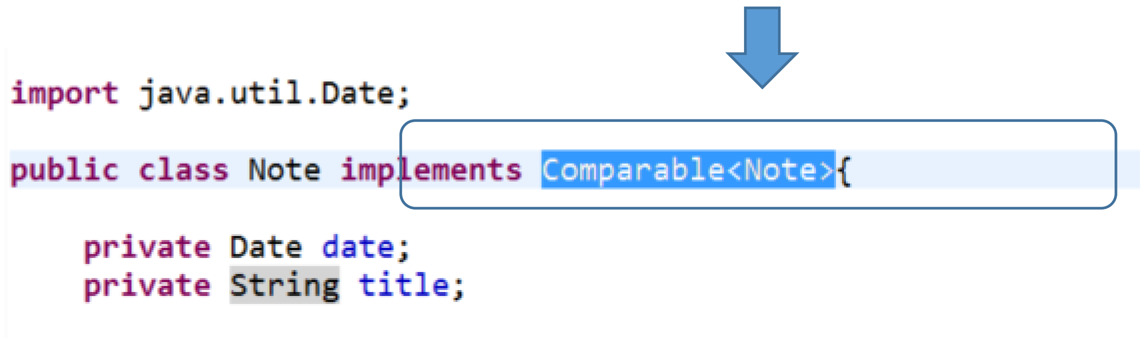


Background

- **Comparable Class**

- In order to make the instances of a class comparable, you need to implement the interface of Comparable for the class

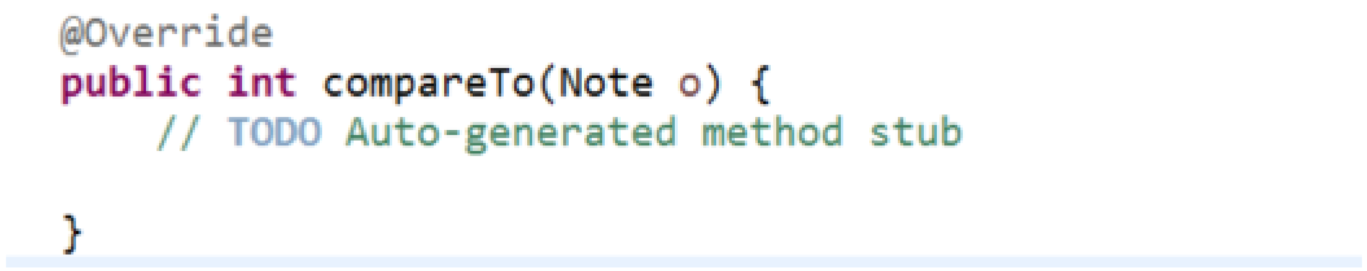
For example:



```
import java.util.Date;

public class Note implements Comparable<Note>{

    private Date date;
    private String title;
```

- **Comparable Class**

- Then you need to override the function compareTo() for the class to specify the comparing rules



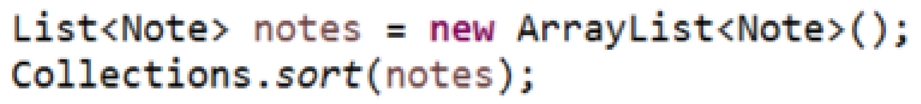
```
@Override
public int compareTo(Note o) {
    // TODO Auto-generated method stub

}
```

The ***compareTo***(Note o) method compares this note with the input Note o.

1. Return 1 means this object is greater than object o.
2. Return -1 means this object is less than object o.
3. Return 0 means these two objects are equal.

• Comparable Class

- Since instances of class Note are comparable, we can sort a collection of Notes by the following:



```java
List<Note> notes = new ArrayList<Note>();
Collections.sort(notes);
```

- Collections.sort() sorts the list of objects from the smallest to the largest.

Your Lab Task

- **Tasks**

Make Class Note, Folder Comparable

Implement the interface Comparable and override compareTo() for both the two classes.

Create searchNotes() function for Class Folder and TextBook

Return a list of Notes which satisfy the searching criterion.

Test Your Program

Download the **`testLab3.java`** from the course website.
Import to your project.
Run this program and show your results to TAs.

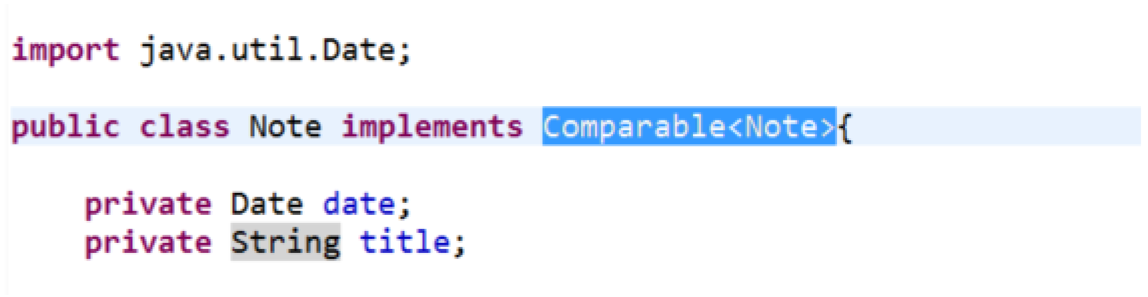
- **Step #1: Make Class Comparable**

  1. Open Eclipse, go to the project *comp3021lab* you created last time.

  2. Find class Note and Folder

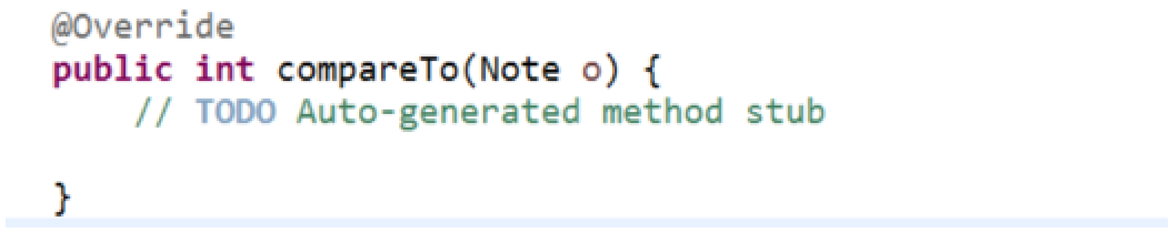
- **Step #1: Make Class Comparable**

1. Implement the interface Comparable



```
import java.util.Date;

public class Note implements Comparable<Note>{

    private Date date;
    private String title;
```

2. Override method compareTo()



```
@Override
public int compareTo(Note o) {
    // TODO Auto-generated method stub

}
```

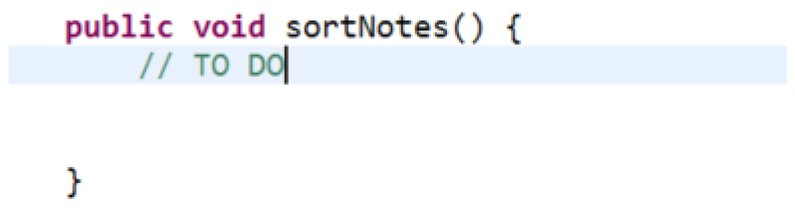
You can specify your own rule for comparing two objects.
In this lab:
For class Note, we compare it based on its creation date, note created more recently is considered as smaller in this lab.
For class Folder, we compare its name. Folder with smaller name is considered as smaller.

You can refer to String.compareTo() to compare two Strings;

- **Step #2: Sort Notes**

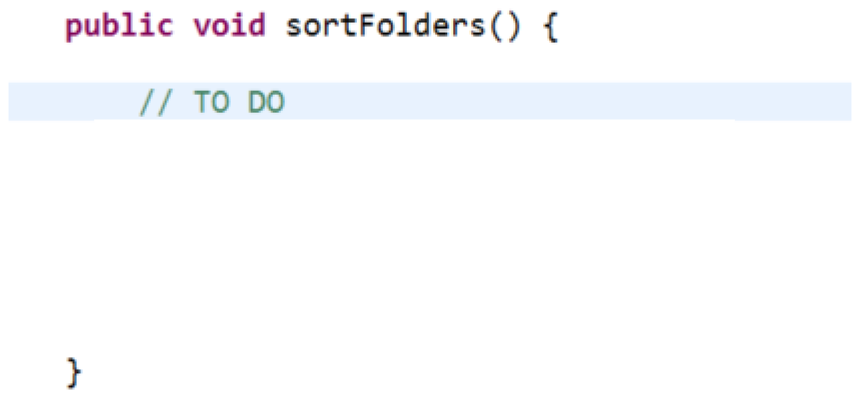
1. Create function sortNotes() for class Folder

   This function sorts all the notes for the folder. You can use `Collections.sort()` to finish this.



```
public void sortNotes() {
    // TO DO


}
```

2. Create function sortFolders() for class NoteBook



```
public void sortFolders() {

    // TO DO




}
```

This function first sorts the notes for each of the folder.
You can leverage Folder.sortNotes() to finish this.

It then sorts all the folders for the note book.

- **Step #3: Search Notes**

1. Create function searchNotes() for class Folder

   This function takes in input a String of keyword separates by a blank space, it returns a list of Notes which contain the keywords specified.

```
public List<Note> searchNotes(String keywords) {

    // TO DO

}
```

This format of keywords is

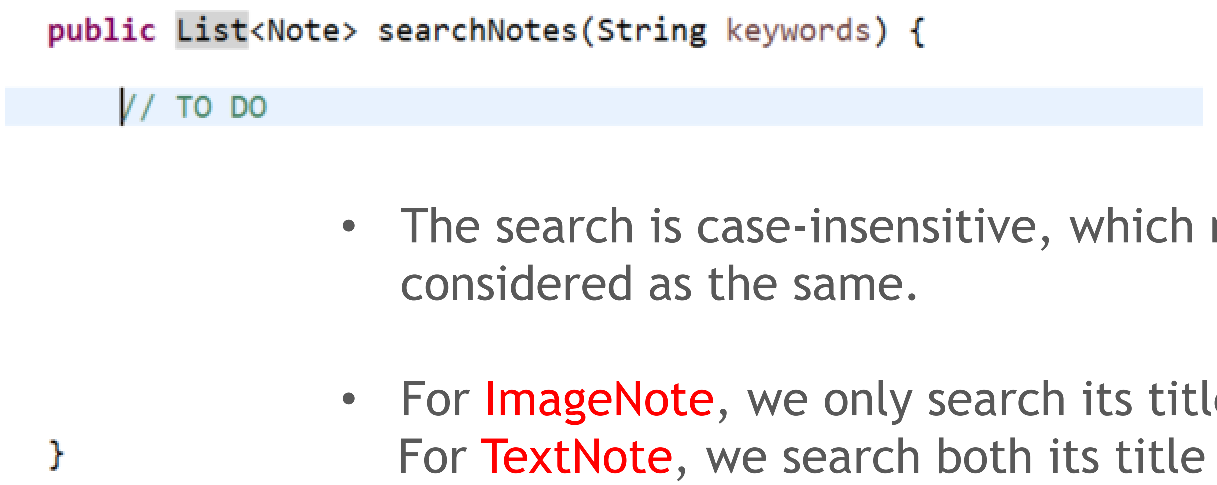
“key1 key2 OR key3 key4” ,
 it means “key1 AND (key2 OR key3) AND key4”

For example: if the search keywords is “java or LAB attendance OR SESSION”
It means that we want to search the notes which contain “java” or “lab” AND contain “attendance” or “session” at the same time.

Step #3: Search Notes

1. Create function searchNotes() for class Folder

   This function takes in input a String of keyword separated by a blank space, it returns a list of Notes which satisfy the searching criterion.



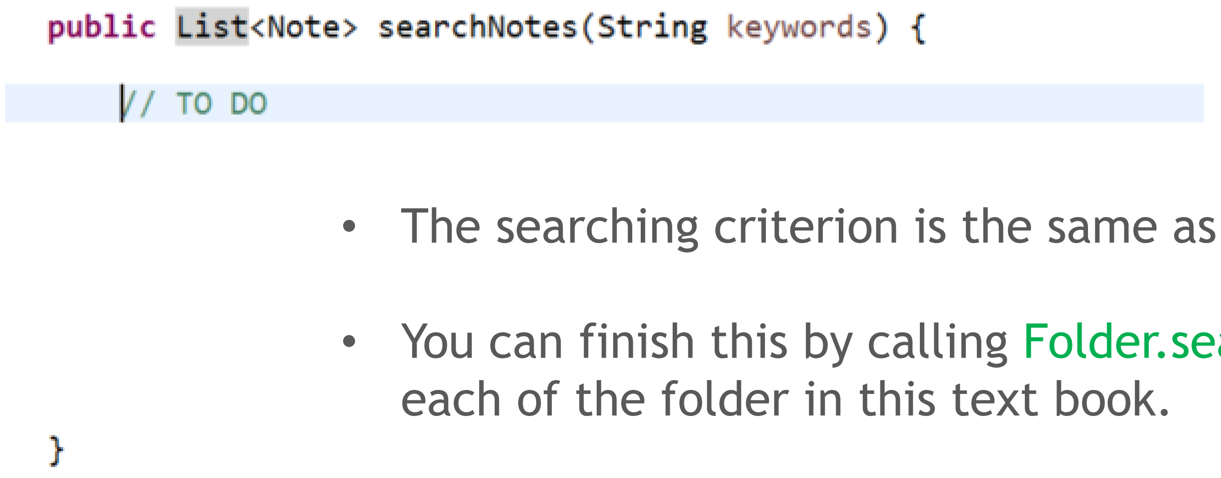
```
public List<Note> searchNotes(String keywords) {

    // TO DO

}
```

- The search is case-insensitive, which means “lab” and “LAB” are considered as the same.
- For ImageNote, we only search its title.
  For TextNote, we search both its title and its contents.
- Only “or” and “OR” are considered as operations, other words are considered as searching keywords.

- **Step #3: Search Notes**

1. Create function searchNotes() for class NoteBook

   This function searches all the notes in all the folders for a Note Book.



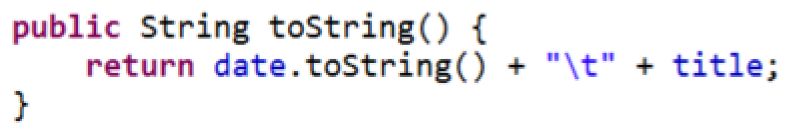
```
public List<Note> searchNotes(String keywords) {
    // TO DO

}
```

- The searching criterion is the same as defined before.
- You can finish this by calling Folder.searchNotes(String Keywords) for each of the folder in this text book.

- **Step #4: Implement other functions**

1. Create function toString() for class Note

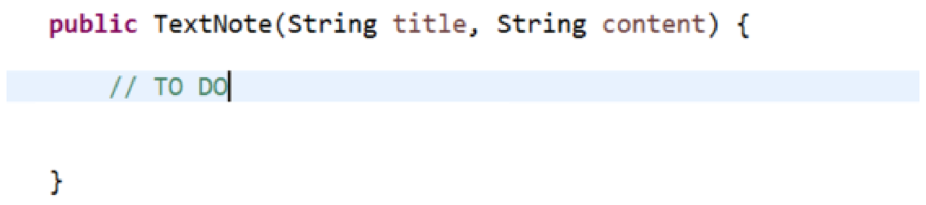
   It prints our the information of date and title for each note.



```java
public String toString() {
    return date.toString() + "\t" + title;
}
```

2. Create another constructor for class TextNote.

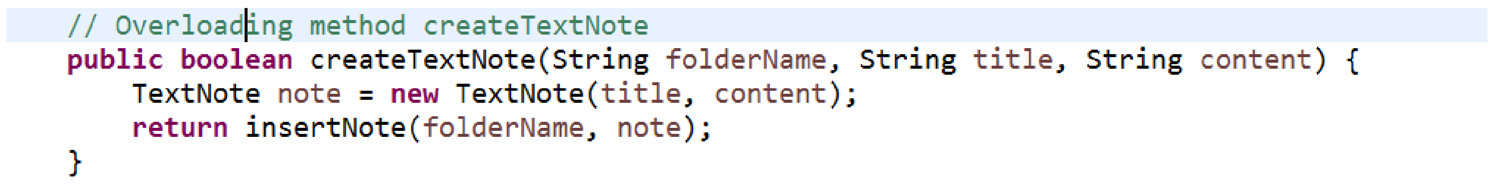
   This constructor can also initialize the content.



```java
public TextNote(String title, String content) {

    // TO DO


}
```

3. Overloading function createTextNote() with context for class Text Note

   This function enables to insert a TextNote with content

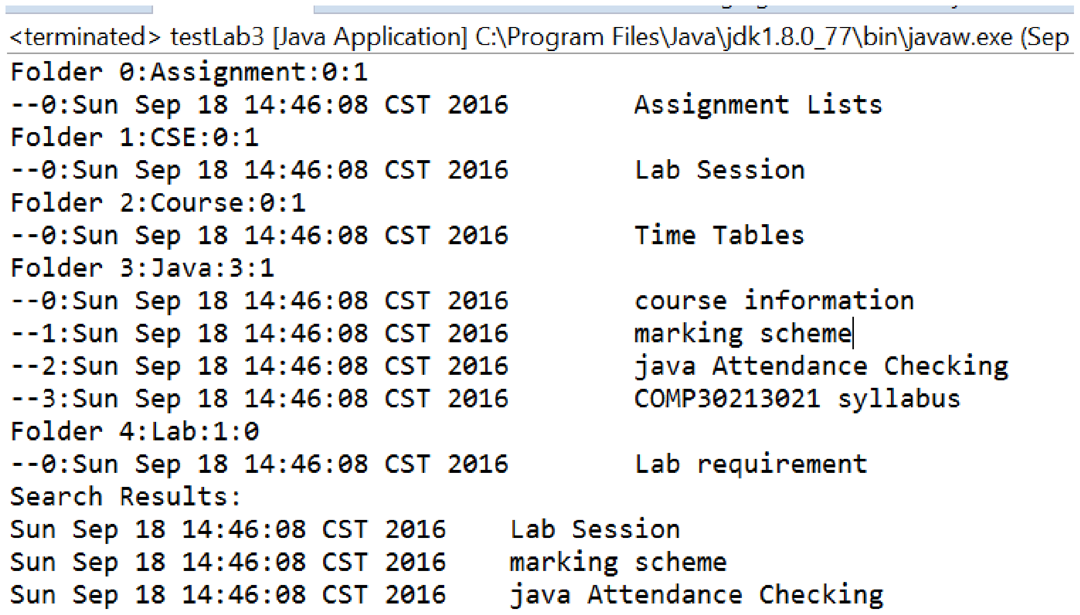


```java
// Overloading method createTextNote
public boolean createTextNote(String folderName, String title, String content) {
    TextNote note = new TextNote(title, content);
    return insertNote(folderName, note);
}
```

- **Step #5: Test**

1. Download the `testLab3.java` from the course website.
2. Import it to your project.
3. Run this program and show your results to TAs.

Expected Output:



```
<terminated> testLab3 [Java Application] C:\Program Files\Java\jdk1.8.0_77\bin\javaw.exe (Sep
Folder 0:Assignment:0:1
--0:Sun Sep 18 14:46:08 CST 2016        Assignment Lists
Folder 1:CSE:0:1
--0:Sun Sep 18 14:46:08 CST 2016        Lab Session
Folder 2:Course:0:1
--0:Sun Sep 18 14:46:08 CST 2016        Time Tables
Folder 3:Java:3:1
--0:Sun Sep 18 14:46:08 CST 2016        course information
--1:Sun Sep 18 14:46:08 CST 2016        marking scheme
--2:Sun Sep 18 14:46:08 CST 2016        java Attendance Checking
--3:Sun Sep 18 14:46:08 CST 2016        COMP30213021 syllabus
Folder 4:Lab:1:0
--0:Sun Sep 18 14:46:08 CST 2016        Lab requirement
Search Results:
Sun Sep 18 14:46:08 CST 2016    Lab Session
Sun Sep 18 14:46:08 CST 2016    marking scheme
Sun Sep 18 14:46:08 CST 2016    java Attendance Checking
```

END OF LAB #3

Don’t forget to commit and push your code.