Collection 补充

1. Collection 类的可嵌套性

在 Collection 类实例化之后，得到的就是一个一般的对象，因此 Collection 类中的数据结构都是支持多层嵌套的。

```
List<List<List<String>>> list = ... // valid

HashMap<String, List<String>> map = ... // valid
```

需要注意的是，如上的声明只是表示了最外层的数据结构，类似于数组中

`int [][][] arr = new int [3][][]` 的做法。内部还是需要自己填充的。

2. 在 IDEA / Eclipse 中调试 TreeMap / TreeSet

TreeMap / TreeSet 虽然内部采用树实现，但是 IDE 为了方便程序员的调试，隐藏了内部的真实实现结构，因此有些同学们可能在调试的时候无法看到。

对于使用 Eclipse 的同学，在 `Variable` 页右上角有一个 `Show Logical Structure` 的按钮，点击关闭即可。

见此链接：https://stackoverflow.com/questions/24738759/eclipse-treemap-debug-view

对于使用 IDEA 的同学，打开设置，找到 `Enable alternative view for Collections classes` 设置项，取消勾选即可。

见此链接：https://stackoverflow.com/questions/52448351/why-dont-i-see-all-variables-in-debugger

3. 自动生成 equals() / hashCode()

在 Eclipse 中，选择 `Sources` 菜单，下有 `Generate equals() and hashCode()` 功能。

在 IDEA 中，先进入某一个类的编辑界面，再右键 ->`Generate...`->`equals() and hashCode()`，或是保持光标在类的代码中，按下 `Alt+Insert`，选择 `equals() and hashCode()`。

注意:自动生成并不是万能的，如果要保证程序的正确性，依然要遵循课上讲过的原则。