**CS 1302 – Test 4 Expectations**

**Information about the test**

General

* Test 4 covers Ch 6, 7, & 8
* You can begin the test approximately 10 minutes before the official beginning of class. When you enter the room, leave everything (all books, phones,smart watches, *etc*)at the front of the room.
* It will be mostly writing classes, methods, and snippets.

**Ch 6 Expectations**

* 1. Given the description of a class, write a method or constructor that throws a *RuntimeException* under certain circumstances.
  2. Given a class with a method or constructor that might throw an exception, create an instance of the class, printing any exception that might be thrown. Hint: you should also make sure that the object that is created in the process is available after the *try/catch* block.
* See: (a) Section 6, and (b) Section 6.1, Exercise 1

1. Know the rule for determining if a thrown exception is caught by a particular *catch* block.

* See: (a) Section 4, (b) Section 4.1, Exercises 1-2.

1. Trace code that utilizes try/catch/finally.

* See: (a) Section 2, (b) Section 7

**Ch 7 Expectations**

1. Given the description of classes involved in a one-to-many relationship, and the description of a text file that whose contents correspond to various objects of these classes, write code that reads such a text file and builds a one-to-many object model in memory.

* See: (a) Section 7.1, (b) Section 7.3, (c) Section 7.4, Exercise 2, (d) HW 6 except without dealing with invalid data.

1. Write code that splits a string on a delimiter(s).

* See: Section 7

**Ch 8 Expectations**

1. Know how to create and use: *ArrayList, HashSet* and *TreeSet*

* See: (a) Section 2, (b) Section 2.1, Exercises 1-4, (c) Sections 6-7, (d) Section 7.1, Exercises 8-10, (e) Section 9.1, Exercises 11-14, (f) Section 10

1. Write code to utilize an *Iterator* to filter a collection by removing items that meet some condition(s), and possibly returning those removed items in a list.

* See: (a) Section 3.1-3.3, (b) Section 3.4, Exercise 1

1. Write a *Comparator* and use it to sort a collection or create a *TreeSet.*

* See: (a) Section 5, (b) Section 5.1, Exercise 6

**Miscellaneous**

You will be provided a handout with this class diagram (next page):

