ITIS/ITCS 4180/5180 Mobile Application Development
In Class Assignment 1

Basic Instructions:

1. In every file submitted you MUST place the following comments:
   a. Assignment #.
   b. File Name.
   c. Full name
2. Each student is required to submit the assignment on Moodle.
3. Please download the support files provided with this assignment and use them when implementing your project.
4. **Export your project as follows:**
   a. From eclipse, choose *"Export…"* from the File menu.
   b. From the Export window, choose *General* then *File System*. Click *Next*.
   c. Make sure that your project for this assignment is selected. Make sure that all of its subfolders are also selected.
   d. Choose the location you want to save the exported project directory to. For example, your *Desktop* or *Documents* folder.
   e. When exporting make sure you select *Create directory structure for files*.
   f. Click Finish, and then go to the directory you exported the project to. Make sure the exported directory contains all necessary files, such as the .java and resource files.
5. Submission details:
   a. All the group members should submit the same zip file.
   b. The file name is very important and should follow the following format: **800#_InClass01.zip**
   c. You should submit the assignment through Moodle: Submit the zip file.
6. **Failure to follow the above instructions will result in point deductions.**

In Class Assignment 1 (100 Points)

In this assignment you will practice using Data Structures and Object Oriented concepts in Java. ***Your implementation should target the most efficient algorithms and data structures. You will be graded based on the efficiency of your implementation. You will not be awarded any points if you use simple nested loops to implement the below tasks.*** You should use one or more of the below data structures:

- ArrayList :
  - JavaDoc: http://docs.oracle.com/javase/7/docs/api/java/util/ArrayList.html
  - Tutorial: http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html
- HashSet :
  - JavaDoc: http://docs.oracle.com/javase/7/docs/api/java/util/HashSet.html
  - Tutorial: http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html
- HashMap :
  - JavaDoc: http://docs.oracle.com/javase/7/docs/api/java/util/HashMap.html
  - Tutorial: http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Question (100 Points)

You are given a file "topmovies.csv"[1] which contains the top-grossing movie for each year, based on tickets sold for each movie during the course of the year. Each line of the file represents a single movie record. Each record consists of a movies's year, name, and total for that year ($). The values are comma separated, for example:

2016,Star Wars Ep. VII: The Force Awakens,69802101

You are asked to perform the following tasks:

1. Create an InClassOn.java, which should include the implementation for this question
2. Create "Movie" class that represents the details provided in the file
3. Read the records in the file. You should implement the parseMovie() method in the User class. **Hint**: extract each value from a movie's record using Java's String.split() method and set the delimiter to a comma, see provided code below. Each movie's record should to be assigned to a Movie object.
4. You are asked to print the list of movies ordered by total($) in descending order. **Hint**: To sort use the Collections.sort(). http://docs.oracle.com/javase/6/docs/api/java/util/Collections.html
5. You are asked to print movies grouped by first letter. i.e. :

```
B: Batman
I: Independence Day, Iron Man 3
…
```

Hint: you need to use one (or more) of the data structures listed above.

[1] source: http://www.the-numbers.com/market/

Some Sample Code

The following code reads in a file line by line. It is assumed that the file is included in root folder of the Eclipse project. Use this code to help you read the provided files.

```
public void readFileAtPath(String filename) {
	// Lets make sure the file path is not empty or null
	if (filename == null || filename.isEmpty()) {
		System.out.println("Invalid File Path");
		return;
	}
	String filePath = System.getProperty("user.dir") + "/" + filename;
	BufferedReader inputStream = null;
	// We need a try catch block so we can handle any potential IO errors
	try {
		try {
			inputStream = new BufferedReader(new FileReader(filePath));
			String lineContent = null;
			// Loop will iterate over each line within the file.
			// It will stop when no new lines are found.
			while ((lineContent = inputStream.readLine()) != null) {
				System.out.println("Found the line: " + lineContent);
			}
		}
		// Make sure we close the buffered reader.
		finally {
			if (inputStream != null)
				inputStream.close();
		}
	} catch (IOException e) {
		e.printStackTrace();
	}
}// end of method
```

String Tokenization:

To split the contents of a single line read a file.

```
String[] resultingTokens = lineContent.split(",");
for (int i = 0; i < resultingTokens.length; i++){
		System.out.println(resultingTokens [i].trim());
}
```