ITIS/ITCS 4180/5180 Mobile Application Development
Homework 1
Date Posted: 08/21/2014 at 14:00
Due Date: 08/24/2014 at 23:55

Basic Instructions:

1. In every file submitted you MUST place the following comments:
    a. Assignment #.
    b. File Name.
    c. Full name of all students in your group.
2. Each group should submit only one assignment. Only the group leader is supposed to submit the assignment on behalf of all the other group members.
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. When you submit the assignment, compress your exported project into a single zip file. The format of compressed file name is HW#.zip
    b. You should submit the assignment through Moodle: Submit the zip file.
6. **Failure to follow the above instructions will result in point deductions.**

Homework 1 (100 Points)

In this assignment you will get familiar with Java's Map, List and Set Interfaces. You will also practice some Object Oriented techniques by creating your own Java class to use within your code. ***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 the Map, List, and/or Set interfaces, and you are encouraged to review the lecture slides and the Java documentation. This assignment consists of 2 parts:

Part 1 (40 Points):

You are given two files: 1) "employeePasswords.txt" which contains the Company X's employees' passwords, and 2) "passwordDictionary.txt" which is a dictionary of common passwords gathered from the Internet. The company's administrator wants to efficiently find out the list of employee passwords that exist in the password dictionary so that he can notify the respective employees and ask them to create a different stronger password. You are asked to perform the following tasks:

1. Create a class called PartOne.java, which should include the implementation for this question. Design the classes following proper Object Oriented programming principles.
2. Read both the provided files and print the list of employee passwords that are present in the password dictionary provided. Your implementation should consider the most efficient (processing and memory) approach to implement this task.

Part2 (60 Points):

In this part, you are given the file "data.csv", which includes live information about medicare offices locations in Australia last updated on August 4th, 2014[1].
Each line in the file represents a medicare office information in the format:

```
id, site_name , address, suburb, state , postcode, longitude, latitude
```

Example record:

```
25,Bowen,53 George Street ,Bowen,QLD,4805,-20.01501,148.24674
```

This file contains 244 records.

You are asked to perform the following tasks:

1. PartTwo.java should include the implementation for this part. Design the classes following proper OO principles to help you perform the following tasks.

[1] Data source https://data.gov.au/dataset/70f76a9e-d450-4567-9760-290dea54229c/resource/7eb30338-768e-436c-9dde-dc7a1066c9c9/download/medicareofficelocationsasat5august2014.csv

2. Read the records in the data.csv file, Hint: extract each value from a MediCare office record using Java's String.split method and set the delimiter to a comma, see provided code below.
3. Each MediCareOffice record needs to be assigned to a MediCareOffice object. To do so, you will create a MediCareOffice class with the required data members and methods.
4. You are asked to efficiently provide the following statistics based on the provided data. You should count how many offices there are in in each state. Print this statistic as a pair of state and count ordered in descending order based on the computed count.

```
────────────────
 State : Count
────────────────
NSW:91
VIC:56
QLD:49
WA:19
SA:13
TAS:7
ACT:5
NT:4
```

Code Snippets

Read File:

The following code that reads in a file line by line. It is assumed 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());
}
```