CS5010, Fall 2022

Assignment 3

Brian Cross and Tamara Bonaci

Assignment from Therapon Skoteiniotis and Tamara Bonaci

- This assignment is due by 11:59pm Monday, October 24th.
- Please push your individual solutions to your designated repo into a branch named "assignment3" within the CS5010 GitHub organization. When you are ready to turn in, create a Pull Request and assign to the TAs.

In general, for all homeworks, please make sure the following is submitted:

- Full project. It must build and run.
- Tests must build and run.
    - Coverage must by 70% for both instruction and Branch 70%
    - Meaningful tests. Don't just write one test case to test multiple methods.
- Javadoc your code
- We'll let the .gitignore do the work of removing files from submit. We want buildable projects.
- Do not include magic numbers.
- Override equals() and hashcode() for public types. Override toString() where It Is helpful and/or makes sense.
- Follow correct coding style, etc.

Do your design first. Use UML as a tool (not just something you generate at the end).

Assignment objectives:

After completing this assignment, you will have gained:

- Increased familiarity with data collections in Java
- Command line input argument parsing
- Java I/O
- Regular expressions

Resources

You might find the following online resource useful while tackling this assignment:

- https://docs.oracle.com/en/java/javase/17/docs/api/
- JUnit 5 Getting Started Online Documentation
- https://docs.oracle.com/javase/tutorial/collections/index.html
- https://docs.oracle.com/javase/tutorial/essential/io/index.html
- https://docs.oracle.com/javase/tutorial/essential/environment/index.html
- https://docs.oracle.com/javase/tutorial/essential/regex/index.html

Gradle

In this assignment, you will continue to Gradle as we used in Lab 1 and Assignment 1. You should be able to copy the build.gradle file from your previous assignment.

Submission Requirements

1. Always create a directory at the root of your repo with the name of the assignment (or lab). example, "**assignment3**".
2. Put all materials for that assignment in that directory.
3. Within that directory, you should have a separate package for every problem.
   a. The package names should follow this naming convention: assignmentN.problemM where you replace N with the assignment number, and M with the problem number, e.g., all your code for problem 1 for this assignment must be in a package named: assignment1.problem1. (There is only one problem in Assignment 3)
4. Make sure to include the build.gradle and other files.
   a. It's easiest to just work within the directory on your machine.
   b. We must be able to load the project, build it, and run it (and your tests)
5. Make sure your programs have a main method and that it is able to run.
6. Please include a **readme.md** file that specifies how to run your code. Especially if parameters are used.

Additional expectations for every package:

- One .java file per Java class
- One .java file per Java test class
- One pdf or image file for each UML Class Diagram that you create
- All methods must have tests
- All non-test classes and non-test methods must have valid Javadoc

Lastly, a few more rules that you should follow in this assignment:

- Your classes should have a `public` modifier
- Instance fields should have a `private` modifier
- Use this to access instance methods and fields inside a class

Code Criteria

- **Gradle built:** Your project should successfully build using the provided **`build.gradle`** file, and it should generate all the default reports.
- **Javadoc generation:** Your Javadoc generation should complete with no errors or warnings.
- **Follow the Java coding conventions:** You should follow the Java coding conventions, especially the naming conventions.
- **Code coverage report:** Your JaCoCo report must indicate 70% or more code coverage per package for "Branches" **and** "Instructions".
- **Methods `hashCode(), equals(), toString():`** all of your classes *have to* provide appropriate implementations. (Appropriate means that it is sufficient to autogenerate these methods, as long as autogenerated methods suffice for your specific implementation). Please don't forget to autogenerate your methods in an appropriate order - starting from the ancestor classes, towards the concrete classes.

- **Javadoc:** please include a short description of your class/method, as well as tags from @params and @returns in your Javadoc documentations (code comments). Additionally, if your method throws an exception, please also include a tag @throws to indicate that.

- **UML diagrams:** Please include UML diagrams for the final versions of your designs for every problem. In doing so, please note that you do not have to hand-draw your UML diagrams anymore. Auto-generating them from your code will be sufficient.

ASSIGNMENT 3

Problem: Insurance Company Email Automation

You are a member of an IT team for a nationwide insurance company. Unfortunately, your company recently suffered a devastating data breach. Private and sensitive data of millions of your customers may now be compromised…

The whole company is still grappling with the consequences of the breach, but you know the drill – it is time to inform the customers. The members of the board of the company have decided to send an email informing every customer about the data breach. Then, the company will also send the same information using the regular mail. **They are asking you to help them out, and to automate the process that the insurance company will use to communicate with its customers.**

The CSV File

The insurance company holds some information about their members in a CSV file[1]. CSV file is organized as a plain text file, containing information such that each piece of data is enclosed in double quotes, and separated by a comma. The first line of the file contains the headers for each column.

```
"first_name" , "last_name" , "company_name" , "email"
"James"      , "Reign"      , "Benton",   "james.reign@gmail.com"
"Josephine"  , "R, Darakjy", "Chanay",    "josie55@hotmail.com"
"Art"        , "Venere"    , "Chemel",    "art2smart@gmail.com"
```

For example, the preceding listing has 4 columns named **`first_name`**, **`last_name`**, **`company_name`** and **`email`**. The second line has the information for member **James Reign**.

Note 1: while the information is enclosed in double quotes, and separated by comma, within the double quotes, there may exist column entries that contain comma. For example, "**R, Darakjy**" is one valid piece of information, and not two.

A sample file, with some of the insurance company's members information, is available on Piazza, and titled insurance-company-members.csv. The CSV file contains first and last name,

[1] This partially explains why they suffered the data breach ☺

company name, address, city, county, state, zip, phone1 and phone2, email address and web page URL.

Given this CSV file, the insurance company would like you to create a program that they can run on the command line. The program should take this file as input, and generate files that will contain the email messages and letters to send to their members. Email messages and letters will be generated from **templates**.

The templates are stored as text files with special placeholders in the text that refer to the CSV file's header names. Here are two example templates, one for email and one for letters. Placeholders are placed between [[and]].

```
To:[[email]]
Subject: Insurance company – information about recent data breach
Dear [[first_name]] [[last_name]],
As you may have heard or read, last month we learned that criminals
forced their way into our systems, and stole information about our
customers. Late last week, as part of our ongoing investigation, we
learned that the taken information includes names, mailing addresses,
phone numbers or email addresses.
I am writing to make you aware that your name, mailing address, phone
number or email address may have been taken during the intrusion.
I am truly sorry this incident occurred, and I sincerely regret any
inconvenience it may cause you.
Because we value you as a customer, and because your trust is
important to us, our company is offering you one year of free credit
monitoring through Experian’s ProtectMyID product, which includes
identity theft insurance where available. You will receive more
information about this offer via regular mail.
You can find additional information and FAQs about this incident at
our website. If you have further questions, you may call us at 866-
852-8680.
Thank you for your patience and your loyalty.
Sincerely,
Insurance Company CEO
```

So, given the above email template and the following line from the CSV file:

```
"first_name","last_name","company_name","address","city","county","state","zip","phone1","phone2","email",
“website”
"Art","Venere","Chemel, James L Cpa","8 W Cerritos Ave #54","Bridgeport","Gloucester",

"NJ","08014","856-636-8749","856-264-4130","art@venere.org","http://www.chemeljameslcpa.com"
```

the email message that is generated looks like:

```
To:art@venere.org
Subject:Insurance company – information about recent data breach
Dear Art Venere,
As you may have heard or read, last month we learned that criminals
forced their way into our systems, and stole information about our
customers. Late last week, as part of our ongoing investigation, we
learned that that information includes names, mailing addresses, phone
numbers or email addresses.
I am writing to make you aware that your name, mailing address, phone
number or email address may have been taken during the intrusion.
I am truly sorry this incident occurred, and I sincerely regret any
inconvenience it may cause you.
Because we value you as a customer, and because your trust is
important to us, our company is offering you one year of free credit
monitoring through Experian’s ProtectMyID product, which includes
identity theft insurance where available. You will receive more
information about this offer via regular mail.
You can find additional information and FAQs about this incident at
our website. If you have further questions, you may call us at 866-
852-8680.
Thank you for your patience and your loyalty.
Sincerely,
Insurance Company CEO
```

Similarly, we also have a template for the company's letter:

```
[[first_name]] [[last_name]]
[[address]], [[city]],
[[county]], [[state]], [[zip]]
([[email]])

Dear [[first_name]] [[last_name]],
As you may have heard or read, last month we learned that criminals
forced their way into our systems, and stole information about our
customers. Late last week, as part of our ongoing investigation, we
learned that the taken information includes names, mailing addresses,
phone numbers or email addresses.
I am writing to make you aware that your name, mailing address, phone
number or email address may have been taken during the intrusion.
I am truly sorry this incident occurred, and I sincerely regret any
inconvenience it may cause you.
Because we value you as a customer, and because your trust is
important to us, our company is offering you one year of free credit
monitoring through Experian’s ProtectMyID product, which includes
identity theft insurance where available. Enclosed please find your
additional information about your free credit card monitoring.


You can find additional information and FAQs about this incident at
our website. If you have further questions, you may call us at 866-
852-8680.
Thank you for your patience and your loyalty.
```

```
Sincerely,

Insurance Company CEO
```

For the same CSV lines we used before, this template generates the following text file:

```
Art Venere
8 W Cerritos Ave #54, Bridgeport,
Gloucester, NJ, 08014.
(art@venere.org)

Dear Art Venere,

As you may have heard or read, last month we learned that criminals
forced their way into our systems, and stole information about our
customers. Late last week, as part of our ongoing investigation, we
learned that the taken information includes names, mailing addresses,
phone numbers or email addresses.
I am writing to make you aware that your name, mailing address, phone
number or email address may have been taken during the intrusion.
I am truly sorry this incident occurred, and I sincerely regret any
inconvenience it may cause you.
Because we value you as a customer, and because your trust is
important to us, our company is offering you one year of free credit
monitoring through Experian's ProtectMyID product, which includes
identity theft insurance where available. Enclosed please find your
additional information about your free credit card monitoring.


You can find additional information and FAQs about this incident at
our website. If you have further questions, you may call us at 866-
852-8680.
Thank you for your patience and your loyalty.
Sincerely,

Insurance Company CEO
```

Note 2: in this assignment, you are provided with **one** example CSV file, and **two** examples of templates. The insurance company, however, may in the future like to write more templates, and your program should accommodate for new templates. **Therefore, your code should work for any CSV file and any template that uses the CSV file's header values as placeholders.**

Command Line Arguments

Your program should accept certain arguments at the command line.

```
--email                          only generate email messages
```

```
--email-template <file>  accept a filename that holds the email
template

--letter                 only generate letters
--letter-template <file> accept a filename that holds the email
template

--output-dir <path>      accept the name of a folder, all output
is placed in this folder

--csv-file <path>        accept the name of the csv file to
process
```

Please note that some options take arguments, for example `--email-template` takes one argument, and it is the name of a file. Similarly, `--output-dir` takes one argument, and it is the name of a folder.

Other options, however, take no arguments, and indicate an action, i.e., `--email indicates` that we should generate emails on this execution of the program.

The command line option `--output-dir` and csv-file are required.

Your program should be able to generate one of the two options (emails or letters) per invocation. If `--email` is given, then `--email-template` must also be provided, if `--letter` is given then `--letter-template` must also be given.

Calling your program and passing any other combination of options should generate an error, for example:

`--email --letter-template letter-template.txt --output-dir letters/` is illegal.

When a user provides an illegal combination of inputs, the program should exit with a helpful error message, and a short explanation of how to use the program along with examples. For example, passing:

```
--email --letter-template letter-template.txt --output-dir
letters

Error: --email provided but no --email-template was given.

 Usage:
   --email                       only generate email messages
   --email-template <file>  accept a filename that holds the
email template.
```

```
   Required if --email is used

        --letter                       only generate letters
        --letter-template <file> accept a filename that holds
the email template.
          Required if --letter is used

        --output-dir <path> accept the name of a folder, all
output is placed in this folder

        --csv-file <path> accept the name of the csv file to
process

Examples:

       --email --email-template email-template.txt --output-dir
emails --csv-file customer.csv

       --letter --letter-template letter-template.txt --output-
dir letters --csv-file customer.csv
```

Finally, please note the order of the command line options does not matter, i.e., the following examples are valid:

```
        --email --email-template email-template.txt --output-dir emails --csv-file customer.csv
        --csv-file customer.csv --email-template email-template.txt --output-dir emails --email
        --output-dir emails --email --csv-file customer.csv --email-template email-template.txt
```

Your tasks:

1. Design and implement the email and the letter generator program for the insurance company, as described above.
2. Please write appropriate Javadoc documentation for all your classes and methods.
3. Please write the corresponding test classes for all your classes.
4. Please use **insurance-company-members.csv**, **email-template.txt** and **letter-template.txt** to help you develop and test your code.
5. Please also develop your own templates (one for email, and one for letter), and include them with your submission.
6. Please provide a final UML Class Diagram for your design

Note 3: Please also make sure that your program works correctly regardless of how your operating system represents paths and files. Make sure not to include hardcoded paths to your files that will only work on your computer.