44-642 Exam 2 - Take Home

Important: Copying and pasting code is *not allowed* (except as noted in the instructions). All code should have comments throughout, unique for each student. Code that is too similar in formatting, spacing, and variable names may be considered plagiarism/academic dishonesty. Bottom line, do your own work and you should be fine.

Problem 1 - Maven

1. Create a Maven project in NetBeans using the `maven-archetype-quickstart` template, with **Group Id:** `com.lastname.firstname` and **Artifact Id:** `exam2p1`.
2. Use `search.maven.org` and/or `mvnrepository.com` to find three different artifacts that you will demonstrate in your project.
    a. Each artifact you use should be in a different category (e.g. cloud, JSON, regex, etc.). Hint: You can search by category at mvnrepository.com. DO NOT USE any artifact we have already used in a project in class assignments or demos.
    b. Make sure you can find enough documentation as to what each artifact does, so you are actually able to use its API in your project. There might be a github repo or other website with the documentation – you may have to research this. If you cannot find sufficient documentation, choose another artifact.
3. Add the dependencies for all three chosen artifacts to your project.
4. Demonstrate at least at least 2 functionalities from each artifact in your program (use whatever imports, methods, and objects from each module as necessary). Show your results as output to the screen (if your demo includes file manipulation, state what happened as part of your program output, and include the result file(s) in your project's directory for submission).
5. Fully comment your program so someone can look at the code and understand quickly what it is about. Describe the purpose of each piece of code you use from the artifacts.
6. It should be obvious what your program is trying to do for anyone who reads the output. Suggestion: label each section of your output with the name of the artifact being demonstrated.

Problem 2 – Factory Patterns

1. Create a new NetBeans project (using Ant) titled `YourNameExam2P2`.
2. Think of a domain/subject area that interests you, and create a simulation using the Abstract Factory Pattern discussed in class (refer to AbstractFactoryPattern ppt from Week 6). If you like the idea you posted in the "Factory Activity Ideas" discussion, feel free to use that subject. Your scenario just has to make sense as an application using abstract factory and the related pieces of the template/pattern (see below).
3. Your simulation should include, at minimum:
    a. A driver class.
    b. A factory producer class with a method that returns the appropriate factory.
    c. An abstract factory interface with at least two abstract methods.
    d. At least two factory classes that implement the abstract factory.
    e. At least two other classes per factory from (d) that are produced by those factories. These classes should also share interfaces as appropriate (see the PowerPoint notes for examples). These classes should also have instance variables.
4. Additional requirements:
    a. Your program should provide a menu and accept user input, looping until the user quits.

   b. Your menu should have some sort of welcome message that names your company or some other label that illustrates your subject (e.g. “Welcome to Cloud Services”).
   c. The user should have several options to request the production of different objects, and be able to enter some sort of relevant value(s) for that object that sets an instance variable.
   d. After the user selects an item and enters parameter(s), the program should print a labeled representation of the item with any instance variable values (i.e. use an appropriate toString method).
   e. The program loops until the user quits.
5. As usual, fully comment your code, and it should be obvious what your program does from the output.

Problem 3 – Regex

1. Create a new NetBeans project (using Ant) titled `YourNameExam2P3`.
2. Pretend you are a malware author and you are trying to come up with random-looking web domain names for your command-and-control viruses, e.g. “RDf-45dlfjxg7.com”.
3. First, you need to come up with some original criteria that define your domain name pattern. For example, “it should be 13 characters long, include 3 integers only, no lower case letters, and possibly end in .net or .edu”. You should come up with your own ORIGINAL pattern, of course.
4. Then, write an algorithm that is able to generate a list of random domain names based on your pattern. Make sure you are able to create at least 10 unique domain names.
5. Next, come up with the regex that fits your pattern. When you check the domain names created by your algorithm, they should all be matched by your regex.
6. For the output of your program, you should test your regex against 10 domain names that DO NOT match, as well as 10 domains that were generated by your algorithm and DO match. We should see labels of “Valid” and “Invalid” accordingly, next to each sample domain name.
7. The pattern you come up with is up to you, but at minimum:
   a. It should end with at least two possible dot-somethings, e.g. “.com” and “.net”.
   b. Before the dot, your domain should have at least 10 characters.
   c. Must include letters, numbers, and possibly dashes, proportion and pattern up to you.
8. Again, fully comment your code, and it should be obvious what your program does from the output.

Bonus (up to 5 points)

Come up with a simple and ORIGINAL scenario that demonstrates multithreading (just use two threads). Create a project named `YourNameExam2Bonus` that first demonstrates a race condition using this scenario, then make it thread safe and show what the execution would look like after your fix.

Submit your solution by following the steps below:

- Save your files in NetBeans.
- Zip all projects into one file. (It should be called YourNameExam2 where YourName is your first and last name.)
- Submit the zip file to the Exam 2 Take Home drop box.
- Download the zip file which you have submitted.
- Look into the zip file and verify the relevant files are correct. If not, resave your project in NetBeans and resubmit.