Lab 8: Writing in the Discipline (WiD) assignment

Every major at Dickinson incorporates a requirement known as *writing in the discipline*, or *WiD*. Dickinson’s computer science major satisfies the WiD requirement by including WiD assignments in several different courses. In each of those courses, students must archive their WiD assignment in a *WiD repository*, which is hosted on GitHub. Each WiD assignment incorporates feedback from the instructor and revision by the student. Therefore, the archive of each assignment in the WiD repository will contain at least three parts: (i) an initial version of the assignment; (ii) feedback from the instructor based on the initial version; (iii) a final, revised version of the assignment.

In this lab, you will design and write Java code as a form of writing in the discipline. You will archive an initial and final version of your code together with the instructor feedback. As stated on the course syllabus, our specific WiD goal is to “develop the ability to write code that is readable, maintainable, extensible, and reusable by future developers.”

Assignment instructions

Write a Java program that implements an appointment checking system. Appointments can be of three types: *one-time* appointments, *daily* appointments, or *monthly* appointments. Your program must include a method occursOn(int year, int month, int day) that checks whether the appointment occurs on that date. For example, for a monthly appointment, you must check whether the day of the month matches. For a one-time appointment, you must check if the day, month, and year match. This simple appointment system does not include any notion of when repeating appointments begin: monthly appointments are considered to occur in every month, both past and future; daily appointments are considered to occur on every day, both past and future. Your program can include any number (and type) of classes that you consider appropriate. JUnit tests for all classes are also required. Your work will be graded based on correctness, code clarity, code re-usability, use of polymorphism, comments, and test quality.  
  
After your appointment checking system is complete, write a demonstration class called AppointmentDemo that fills an array of appointment objects with a mixture of appointment types (one-time, daily, monthly). Have the user enter a date and print out all appointments that occur on that date.  
  
A sample interaction between the user and the AppointmentDemo class is as shown below:  
  
Enter a date (like 2015 10 31): **2015 11 1**  
You have the following appointments on 11/1/2015:  
Blood Pressure Check  
Visit to grandma  
Dentist  
Code writing  
  
This lab will not be performed in pairs or teams. Each student will complete the lab individually. The lab will consist of two parts, Part A and Part B. Each part will be submitted and graded separately.

Part A (70 points)

1. **Create Java project, package, and classes:** In this lab, the starter code repository consists of nothing except a single README file. This has been done deliberately so that you can have the experience of creating a completely new project. Here are the steps to achieve this:
   1. Follow the usual steps from the How to… page in order to download the starter code repository, *but* proceed only to the point where the lab-08-yourGithubLogin repository is visible within the Git Repositories view of Eclipse—do *not* attempt to import the project into Package Explorer.
   2. Note the file system location of your lab-08-yourGithubLogin repository (usually visible in light gray at the end of the top line of the Git Repositories view). We will use this location in the next step.
   3. In Package Explorer, right-click and select New | Java Project. Give your project a meaningful name then uncheck the “Use default location” box. In the Location textbox, enter the location of your lab-08-yourGithubLogin repository as recorded in the previous step. Click Finish to complete the creation of your Java project. This creates a src directory and several other files and directories.
   4. Create a new package within your project’s src directory. Right click on src, select New | Package. Give your package an appropriate name and click Finish.
   5. When you are ready to create your first class, right click on your package, select New | Class, and proceed from there.
2. **Write code and submit to GitHub:** Design your appointment system. Write and test the Java code for the system. Aim to produce a complete, polished, well-designed, and fully commented version of the code. Commit and push this to your dickinson-comp132/lab-08-… GitHub repository as for any other lab.
3. **Submit lab report to Moodle:** As for every other lab, submit a lab report to Moodle. The lab report for Part A consists only of a self-assessment report.
4. **Copy all source code and the lab report into your WiD repository:** 
   1. *Create* COMP132 *folder in WiD repository:* The instructor will have created a WiD repository for you if you did not have one already. You will receive an email invitation to collaborate on this repository, and you need to accept this invitation. The repository is located at https://github.com/Dickinson-COMP-WiD/your-dson-login*.* **Please note carefully that the final component of this URL is not your GitHub login; it is your Dickinson login.** Navigate to this repository and create a new folder called COMP132. In fact, it’s not possible to create an empty new folder in GitHub. Instead, you need to create and commit a file in the new folder. Therefore, an easy way to create the new folder is to click on “Add file” then create a new file called COMP132/README.md. This creates the desired new folder and you can also put a descriptive message in the readme file.
   2. *Create* COMP132/PartA *folder in WiD repository:* follow some similar steps to create your PartA folder, again creating a readme file with a descriptive message.
   3. *Copy all source code into* PartA *folder:* Use drag-and-drop to copy the src folder of your Eclipse project into the PartA folder. This results in all your .java files being added to the WiD repository.
   4. *Copy lab report into* PartA *folder:* Again using drag-and-drop, add your lab report to the PartA folder.

All the above steps must be completed before the Lab 8a deadline specified on the class schedule.

The instructor will grade your work and provide feedback. The instructor will also place the feedback into your WiD repository, in a folder called PartA/feedback.

Part B (30 points)

1. **Update code and submit to GitHub:** Taking account of the feedback provided by the instructor, update your Java code and tests. This final version should meet professional standards for software development. Commit and push this to your dickinson-comp132/lab-08-… GitHub repository as for any other lab. Note that this will overwrite your previous Part A code. Also note that if your Part A submission earned a particularly high grade, the instructor may have added an additional requirement that you write some new code. This is done to ensure that you must make a meaningful revision to your initial version of the code. Please read the feedback carefully and take account of all suggestions.
2. **Submit lab report to Moodle:** As for every other lab, submit a lab report to Moodle. The lab report for Part B consists of a special report in which you list each part of the instructor feedback. For each part of the feedback, provide a comment on how you addressed it. No separate self-assessment report is required.
3. **Copy all source code and the lab report into your WiD repository:** 
   1. *Create* COMP132/PartB *folder in WiD repository:* follow steps as in Part A to create your PartB folder, again creating a readme file with a descriptive message.
   2. *Copy all source code into* PartB *folder:* Use drag-and-drop to copy the src folder of your Eclipse project into the PartB folder. This results in all your .java files being added to the WiD repository.
   3. *Copy lab report into* PartB *folder:* Again using drag-and-drop, add your Part B lab report to the PartB folder.

All the above steps must be completed before the Lab 8b deadline specified on the class schedule.