ICS-33: In-Lab Programming Exam #3

General Rules:

This in-lab programming exam is worth a total of 50 points (half the points of the first two In-Lab exams). It requires you to write a generator function (decorator for iterables), a recursive function, and a class derived using single inheritance. I will supply a short script for calling the functions and methods and printing their results, so that you can check them visually for correctness (and more easily debug them). The end of the script calls **driver.driver()** to test your code using batch self-checks (similar to the ones that I will use for grading purposes).

You will have approximately 110 minutes to work on the exam, after logging in and setting up your computer (so that is about 35 minutes/problem). You will then write, test, and debug the functions/methods: they will be in a project folder named **exam**.

We will test your functions/methods only for correctness; each test you pass will improve your grade; functions/methods that produce no correct results will earn no credit. This means that your functions/methods must define exactly the parameters specified in the descriptions below and must work on all the example arguments; your functions/methods should also work on any other similar/correct arguments. To aid you writing and debugging these methods

1. Write clear, concise, and simple Python code (there are no statement restrictions).
2. Choose good names for local variables.

You do not need to include any comments in your code; but, feel free to add them to aid yourself. We will **not** grade on appropriate names, comments, or Python idioms: only on correctness. You may also call extra **print** methods in your code or use the Eclipse Debugger to help you understand/debug your code.

You may import and use any functions in the standard Python library and the **goody** and **prompt** modules (which will be included in the project file that you will download), unless told otherwise in the problem specification. Documentation for Python's standard library and these modules will be available during the exam. I have written all the standard import statements that I needed; feel free to include other imports, or change the form of the included imports to be simpler to use.

If you are having problems with the operating system (logging on, downloading the correct folder/files, accessing the Python documentation, submitting your solution) or Eclipse (starting it, setting it up for Python, running Python scripts, running the Eclipse debugger, turning on line numbers) please talk to the staff as soon as possible. We are trying to test only your programming ability, so we will help you with these other activities. But, we cannot help you understand, test, or debug your programming errors. I have asked the TAs and Tutors to NOT ANSWER ANY QUESTIONS about the exam itself: read it carefully and look at the test cases for clarification.

You should have a good understanding of the solution to Quizzes #4, #5, and #6 (and the Minimum Number of Stamps problem in the lecture notes). You should be familiar with **for** and **while** loops, the use of **try/except** statements, the **min** and **sorted** functions (and their use of the **key** parameter for determining order), lambdas, comprehensions, ***args** as a parameter specification, using **iter** and **next** on iterables, generator functions, recusive functions, single inheritance, and writing dunder methods. You are free to use/avoid whatever Python language features you are comfortable/uncomfortable with. Grading depends only on whether your functions follow the requirements and work correctly.

Before submitting your program, ensure that it runs (has no syntax errors) and ensure there are no strange/unneeded **import** statements at the top of your file.

Summary:

1 of 2

This exam is worth 50 points. It requires you to write one generator function, one recursive function, and one class (using single inheritance) according to the specifications below. It will be graded as follows. If you solve no problems correctly, you will score at least 0 points; if you solve one problem completely correctly (no matter which one), you will score at least 36 points (72%, a **C-**); if you solve two problems completely correctly (no matter which ones), you will score at least 43 points (86%, a **B**); if you solve all three problems completely correctly, you will score 50 points (100% an **A**).

The module you will download...

- imports functions that are used for testing the code you will write.
- defines functions with reasonably annotated parameter names (you can change the names) and whose bodies are **pass**; thus, they will return **None** and will satisfy no tests.
- defines a class in which you must define the headers and bodies of its methods.
- includes a script that tests these functions/methods individually on different legal inputs, printing the results (sometimes showing extra information). You are also welcome to add and run your own tests inside this script.

  The script tests are followed by batch self-check tests.

The next three sections explain the details of these two functions and the class. You should briefly read all three and decide in what order to try solving them. If you are not making good progress toward a solution, move on to work on a different problems (which you might find easier) and return if you have time.

This information will be provided on the acutal test.

2 of 2