DEBUGGING LAB, DUE: OCTOBER 3, 11:59PM EST

1. Instructions

1.1. **What to look at.** The only references you may use for this lab are:

- The mrjob api reference: `https://mrjob.readthedocs.io/en/latest/`
- Any material on Canvas

In particular, do not use stackoverflow or chegg for two reasons, they often give bad advice.

1.2. **How to collaborate.**

- Write down (in your code) the names of people you collaborate with.
- Cannot look at each other's code. You can collaborate by discussing ideas.

1.3. **Getting the code and submitting the code.** Use update-starter to get the code and submit to gradescope. You can submit multiple times based on autograder responses **Make sure to only commit human-generated files, and no other files into your repository.**

After running update-starter, you should see an **debuglab** directory for this assignment.

1.4. **The Assignment.**

Question 0. Starting from this assignment, code will also be graded based on style. Style includes:

- Proper indentation of code (4 space indent, no tabs, this is checked by autograder).
- Use good, informative variable names. Avoid misleading variable names (e.g., `words = [int(x), float(y)]`).
- Use good comments:
  - The **class** should have a docstring (e.g., `""" this is a docstring """`) as its first line (i.e., right below `class MyClass(MRJob):`) that explains its purpose. See **debuglab/q1/q1.py** for an example.
  - The mapper should have a docstring explaining what exactly it expects as the input and what it is supposed to be yielding. See **debuglab/q2/q2.py** for more details.
  - The reducer should also have a corresponding docstring.
  - Non-obvious chunks of code should have a comment explaining their purpose (what are they trying to accomplish?).
  - Avoid "obvious" comments. E.g., `x = x + 1 # adding 1 to x`

Question 1. This question uses the Orders dataset in the HDFS **/datasets/orders** directory. There are 3 types of file chunks there: some chunks are information about customers (customer id, country), some are about items (description, price, StockCode serves as item id) and some are about orders (InvoiceNo serves as order id). In the customer chunks, there are no duplicated customers, and in the item chunks there are no duplicated items. The class docstring in the file **debuglab/q1/q1.py** describes what the class is trying to accomplish and has some code that tries to do it, but fails spectacularly, using some of the most common types of programming errors. Aside from this file (edit and correct it), also fill in **debuglab/q1/run**. Although you can use python's try/except features during debugging, there should be no "try" or "except" in the submitted code to help autograder check your code (in general, try/except are for errors you do not anticipate, so having that in your assignment is the same as not doing it).

Question 2. This question uses the Cities dataset in the HDFS **/datasets/cities** directory. It is a tab-separated dataset that has information about cities (name, state, county, population, zip codes in the city, and an ID). In the file **debuglab/q2** you will find a file **q2.py** that tries to compute, for every state (in the dataset, Washington D.C. counts as a state), the number of cities it has, the total population in the state, and the maximum number of zip codes that appear in a city (in the state). However, the code has many bugs. Your task is to fix the bugs so that the code runs correctly, using your debugging skills. Also fill in the run script **run** (choose the number of reducers based on your knowlege of US geography). Although you can use python's try/except features during debugging, there should be no "try" or "except" in the submitted code so that autograder can check it.

Your submitted code should have a comment section at the end, where you list all of the bugs and explain what you had to do to fix them. It should look something like this (in terms of style):

```
### Bugs detected and fixed ###
# 1. Changed the import from MRSteveJobs to MRJob
# 2. Added the ‘‘self’’ variable that was forgotten in the definition of the reducer
#
```