Spark Tutorial and Word Count Lab

For this week, there is one tutorial and one lab to complete.  **JUST**the lab (lab1\_word\_count) will be submitted to the autograder.  Below are the instructions.

1. Start the VM - To start the VM, from a DOS prompt (Windows) or Terminal (Mac/Linux), issue the command "vagrant up", while in the custom directory [created](https://courses.edx.org/courses/BerkeleyX/CS100.1x/1T2015/courseware/d1f293d0cb53466dbb5c0cd81f55b45b/920d3370060540c8b21d56f05c64bdda/) for this course.
2. Once the Virtual Machine is running, access the Jupyter web UI for running IPython notebooks by navigating your web browser to "<http://localhost:8001/>" (or "<http://127.0.0.1:8001/>").
3. If you have any running notebooks they **SHOULD BE** shutdown.  Only **ONE** notebook should be run at a time.  Running notebooks have a green icon to the left of the notebook name and green text to the right of the screen that says "Running".  Shutdown running notebooks by clicking the checkbox next to the notebook and then clicking the orange "Shutdown" button.
4. Download the IPython notebooks.  Make sure that the file extension is .ipynb.  If the download adds an extension (e.g. ".txt"), rename the file so that the extension is just .ipynb.
   * Spark Tutorial: <https://raw.githubusercontent.com/spark-mooc/mooc-setup/master/spark_tutorial_student.ipynb>. You can view this tutorial online [here](http://nbviewer.ipython.org/github/spark-mooc/mooc-setup/blob/master/spark_tutorial_student.ipynb).
   * Lab 1 Word Count : <https://raw.githubusercontent.com/spark-mooc/mooc-setup/master/lab1_word_count_student.ipynb>. You can view this lab exercise online [here](http://nbviewer.ipython.org/github/spark-mooc/mooc-setup/blob/master/lab1_word_count_student.ipynb).
5. Upload the IPython notebooks.  This process was explained during "[Setting up the Course Software Environment](https://courses.edx.org/courses/BerkeleyX/CS100.1x/1T2015/courseware/d1f293d0cb53466dbb5c0cd81f55b45b/920d3370060540c8b21d56f05c64bdda/)" in week one.
6. For the **Spark Tutorial**notebook, all you need to do is read the text and run the cells.  You will not need to change anything, but feel free to change cells and see how the output changes.  If you change something and can't get a cell to run properly, re-upload the notebook to Jupyter.  Note that this notebook **SHOULD NOT** be submitted to the autograder.
7. For the **Lab 1 Word Count** notebook, please follow the instructions in the notebook and replace <FILL IN> sections with your solution.  After you confirm that your code passes all of the tests while running in the VM, please export it as a **python file(.py)**and submit it to the autograder server.  The submission process is the same as in week 1.  In the next module, the instructions are provided again for your convenience.
8. An outline of what will be covered in the two notebooks is included below.  Feel free to skip this, as this information is also included in the notebooks.
9. When you have submitted successfully, you can shutdown the VM by issuing the command "vagrant halt".

**Please do not post your programming exercises in publically visible repositories, such as GitHub.**

Learning Apache Spark

This lab is consists of two parts: a Spark Tutorial and a word count exercise.

*The lab is due Jun 12, 2015 at 00:00 UTC. There is a three day grace period for late submissions until Jun 15, 2015 at 00:00 UTC. Submissions after that time will lose 20 points.*

SPARK TUTORIAL

This tutorial will teach you how to use [Apache Spark](http://spark.apache.org/), a framework for large-scale data processing, within a notebook. Many traditional frameworks were designed to be run on a single computer.  However, many datasets today are too large to be stored on a single computer, and even when a dataset can be stored on one computer (such as the datasets in this tutorial), the dataset can often be processed much more quickly using multiple computers.  Spark has efficient implementations of a number of transformations and actions that can be composed together to perform data processing and analysis.  Spark excels at distributing these operations across a cluster while abstracting away many of the underlying implementatation details.  Spark has been designed with a focus on scalability and efficiency.  With Spark you can begin developing your solution on your laptop, using a small dataset, and then use that same code to process terabytes or even petabytes across a distributed cluster.

During this tutorial we will cover:

* *Part 1:* Basic notebook usage and [Python](https://docs.python.org/2/) integration
* *Part 2:* An introduction to using [Apache Spark](https://spark.apache.org/) with the Python [pySpark API](https://spark.apache.org/docs/latest/api/python/pyspark.html" \l "pyspark.RDD" \t "_blank) running in the browser
* *Part 3:* Using RDDs and chaining together transformations and actions
* *Part 4:* Lambda functions
* *Part 5:* Additional RDD actions
* *Part 6:* Additional RDD transformations
* *Part 7:* Caching RDDs and storage options
* *Part 8:* Debugging Spark applications and lazy evaluation

The following transformations will be covered:

* map(), mapPartitions(), mapPartitionsWithIndex(), filter(), flatMap(), reduceByKey(), groupByKey()

The following actions will be covered:

* first(), take(), takeSample(), takeOrdered(), collect(), count(), countByValue(), reduce(), top()

Also covered:

* cache(), unpersist(), id(), setName()

Note that, for reference, you can look up the details of the relevant methods in Spark's [Python API](https://spark.apache.org/docs/latest/api/python/pyspark.html#pyspark.RDD).

WORD COUNT EXERCISE: BUILDING A WORD COUNT APPLICATION

This exercise will build on the techniques covered in the Spark tutorial to develop a simple word count application.  The volume of unstructured text in existence is growing dramatically, and Apache Spark is an excellent tool for analyzing this type of data.  In this exercise, we will write code that calculates the most common words in the [Complete Works of William Shakespeare](http://www.gutenberg.org/ebooks/100) retrieved from [Project Gutenberg](http://www.gutenberg.org/wiki/Main_Page).  The code you write could also be scaled to find the most common words on the Internet.

During this exercise we will cover:

* *Part 1:* Creating a base RDD and pair RDDs
* *Part 2:* Counting with pair RDDs
* *Part 3:* Finding unique words and a mean value
* *Part 4:* Apply word count to a file