HOMEWORK 1: BACKGROUND

10-601 Introduction to Machine Learning (Spring 2018)
Carnegie Mellon University
https://piazza.com/cmu/spring2018/10601
OUT: Jan 17, 2018*
DUE: Jan 24, 2018 11:59 PM
TAs: Brynn Edmunds, Sarah Schultz, Jennifer Shi, Sienna Stritter, Yu Huang

START HERE: Instructions

- **Collaboration Policy**: Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books or online resources, again after you have thought about the problems on your own. There are two requirements: first, cite your collaborators fully and completely (e.g., "Jane explained to me what is asked in Question 3.4"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only. See the collaboration policy on the website for more information: http://www.cs.cmu.edu/~mgormley/courses/10601-s18/about.html

- **Late Submission Policy:** See the late submission policy here: http://www.cs.cmu.edu/~mgormley/courses/10601-s18/about.html

- **Submitting your work:** You will use Canvas to submit answers to all questions, and Autolab to submit your code. Please follow instructions at the end of this PDF to correctly submit all your code to Autolab.

  - **Canvas:** Canvas (https://canvas.cmu.edu) will be used for quiz-style problems (e.g. multiple choice, true / false, numerical answers). Grading is done automatically. You may only **submit once** on canvas, so be sure of your answers before you submit. However, canvas allows you to work on your answers and then close out of the page and it will save your progress. You will not be granted additional submissions, so please be confident of your solutions when you are submitting your assignment. The above is true for future assignments, but this one allows **unlimited submissions**.

  - **Autolab:** You will submit your code for programming questions on the homework to Autolab (https://autolab.andrew.cmu.edu/). After uploading your code, our grading scripts will autograde your assignment by running your program on a virtual machine (VM). The software installed on the VM is identical to that on `linux.andrew.cmu.edu`, so you should check that your code runs correctly there. If developing locally, check that the version number of the programming language environment (e.g. Python 2.7, Octave 4.0.0, OpenJDK 1.8, g++ 4.8) and versions of permitted libraries (e.g. `numpy` 1.7.1) match those on `linux.andrew.cmu.edu`. (Octave users: Please make sure you do not use any Matlab-specific libraries in your code that might make it fail against our tests.) You have a **total of 10 Autolab submissions**. Use them wisely. In order to not waste Autolab submissions, we recommend debugging your implementation on your local machine (or the linux servers) and making sure your code is running correctly first before any Autolab submission. The above is true for future assignments, but this one allows **unlimited submissions**.

- **Materials:** Download from autolab the tar file ("Download handout"). The tar file will contain all the data that you will need in order to complete this assignment.

*Compiled on Thursday 18th January, 2018 at 21:36

1 Prerequisite Practice [75 points]

In this section, you will work through a number of problems covering probability, statistics, calculus, linear algebra, geometry, and computer science. Detailed instructions and the problems themselves are posted on Canvas and should be submitted there.

You can find the assignment at the link below:

`https://canvas.cmu.edu/courses/2666/quizzes/8885`

2 Hello, Autolab! [25 points]

2.1 Introduction

This homework is neither representative of the standard difficulty of programming assignments for this course nor is it designed to test your ability to program. **In this homework you have to choose Python, Octave, Java, or C++ as your programming language. Submitting code for more than one language may result in undefined behavior.**.

The goal of this assignment is to ensure that you:

1. Have a way to edit and test your code (i.e. a text editor and compiler/interpereter)
2. Are familiar with submitting to Autolab
3. Are familiar with file I/O and standard output in the language of your choice

Warning: This handout assumes that you are using a unix command prompt (with `zsh`, `bash`, `csh`, or similar). All of the command prompts lines listed in this handout will work on the linux.andrew.cmu.edu machines. You may need to use other commands or methods if you are working locally – especially if you are using Windows.

2.2 Reading from a file [20 Points]

In `reverse.{py|m|java|cpp}`, implement a program that reads in the lines of a file, then writes them in reverse order to an output file. Specifically, your program should take two command line arguments: the name of the input file and the name of the output file. It should read the lines of the input file and write them to the output file from last to first, separated by `"\n"`. You should assume that the input file has unix-style line breaks[1]

For example, if the file `input.txt` contained the stream

```
#pineapples\n#pinstripes\n#pinwheelofdoom\n#pinsir\n
```

which is commonly displayed as

```
#pineapples
#pinstripes
#pinwheelofdoom
#pinsir
```

depending on your language of choice, one of the following:

- `python reverse.py input.txt output.txt`
- `octave -qH reverse.m input.txt output.txt`
- `javac reverse.java; java reverse input.txt output.txt`

[1] Windows uses `"\r\n"` to indicate a new line. Unix uses only `"\n"`. See Wikipedia for details.

- `g++ reverse.cpp; ./a.out input.txt output.txt`

should write the following to output.txt

```
#pinsir\n#pinwheelofdoom\n#pinstripes\n#pineapples\n
```

which is displayed as

```
#pinsir
#pinwheelofdoom
#pinstripes
#pineapples
```

You may assume that the contents of the input file will fit in memory for any reasonable machine and the contents of the file will be ASCII-encoded. You will be provided with two example files `example.txt` and `sentences.txt` with which you can test your code. However, do not assume that if your code works on just those files then it will receive full points. We will grade your code on different, hidden test cases. Attempts to directly determine the specific contents of these tests constitute violation of the course policy.

Note to Octave users: Please be sure that `reverse.m` is a *script* that gets its arguments from the command line rather than a *function*.

2.3 Test Code on `linux.andrew.cmu.edu` Machines

Before submitting to Autolab on this and **every future assignment**, you should check that it behaves correctly when run on the `linux.andrew.cmu.edu` machines, since they mirror the software installed on the Autolab virtual machines (VMs). These instructions assume you are working on a unix based operating system (e.g. Linux, Mac OS)—if you are using Windows you can install cygwin (https://www.cygwin.com/) which provides a unix-like environment.[2]. (Of course, you are also welcome to develop your code directly on these same servers.)

Follow the three steps below. Here we assume your code and the `example.txt` file are located in a subdirectory `./reverse/`.

1. Copy your code to the linux server:

   ```
   rsync -a ./reverse/ <Your Andrew ID>@linux.andrew.cmu.edu:~/reverse/
   ```

2. Log into the linux server:

   ```
   ssh <Your Andrew ID>@linux.andrew.cmu.edu
   ```

3. Change directories to where you just copied the code:

   ```
   cd ~/reverse/
   ```

4. Run the code with one of the below:

   - `python reverse.py example.txt output.txt`
   - `octave -qH reverse.m example.txt output.txt`
   - `javac reverse.java; java reverse example.txt output.txt`
   - `g++ reverse.cpp; ./a.out example.txt output.txt`

5. Check that it was properly reversed:

   ```
   cat output.txt
   ```

[2] Another alternative for Windows would be to use other GUI tools, WinSCP (https://winscp.net/) and Putty (http://www.putty.org/)

2.4 Autolab Submission [5 Points]

You must submit a .tar file named `reverse.tar` containing `reverse.{py|m|java|cpp}`. You can create that file by running:

```
tar -cvf reverse.tar reverse.{py|m|java|cpp}
```

from the directory containing your code.

Some additional tips: **DO NOT** compress your files; you are just creating a tarball. Do not use tar `-czvf`. **DO NOT** put the above files in a folder and then tar the folder. Autolab is case sensitive, so observe that all your files should be named in **lowercase**. You must submit this file to the corresponding homework link on Autolab.

Note: For this assignment, you may make arbtirarly many submissions to Autolab before the deadline, but only your last submission will be graded.