Assignment 2: Aug 31 - Sept 6 (Basic Python)

Hand-in deadline: Sept 11

Important: The scripts of this assignment should be saved in a subdirectory named `week2` of your INF3331 github repository. Check your Python version with `python --version` and make sure you use Python 2.x.

1.1 Random numbers (2 pts)

Write a script that prints a uniformly distributed random number between -1 and 1 on the screen. The number should be written with four decimals as implied by the `.4f` format.

To create the script file, you can use a standard editor such as Emacs or Vim on Unix-like systems. On Windows you must use an editor for pure text files - Notepad is a possibility, but I prefer to use Emacs or the "IDLE" editor that comes with Python (you usually find IDLE on the start menu, choose File–New Window to open up the editor). IDLE supports standard key bindings from Unix, Windows, or Mac (choose Options–Configure IDLE... and Keys to get a menu where you can choose between the three classes of key bindings).

The standard Python module for generation of uniform random numbers is called `random`. To figure out how to use this module, you can look up the description of the module in the Python Library Reference. Open (google) the Reference in a Web browser and open the `Library Reference` section. You will then see the index of Python functions, modules, data structures, etc. Find the item random (standard module) in the index and follow the link. This will bring you to the manual page for the random module. In the bottom part of this page you will find information about functions for drawing random numbers from various distributions (do not use the classes in the module, use plain functions). Also apply pydoc to look up documentation of the random module: just write `pydoc random` on the command line.

Name of Python scipt: printrandom.py

2.2 Temperatur converter (3pt)

Write a script that converts a temperatur from Fahrenheit to Celsius. The Fahrenheit value should be taken in as a command-line argument. An example usage would be:

```
python temperatureconverter.py 75.2
```

for which the program would output:

```
75.2 Fahrenheit is equal to 24.0 Celsius.
```

Both Fahrenheit and Celsius values should be formated with one decimal.

Name of Python script: temperatureconverter.py

2.3 Line counter (3 pts)

Write a script that counts the lines of a set of files. For example, it can be used with:

```
python linecounter.py file1.txt file2.txt file3.txt
```

for which the program would output:

```
file1.txt: 12
file2.txt: 43
file3.txt: 126
```

where the numbers correspond the the number of lines in these files.

Traverse the command-line arguments using a loop. The complete list of the command-line arguments can be written `sys.argv[1:]` (i.e., the entries in `sys.argv`, starting with index 1 and ending with the last valid index). The for loop can then be written as `for r in sys.argv[1:]`. In the loop, open the file and count the number of lines.

Name of Python script: linecounter.py

2.4 Word counter (2 pts)

Extend exercise 2.3 such that it returns the number of words instead of the number of lines in each file. Otherwise the usage is identical as in exercise 2.3. Use the `split()` function to break down a line into individual words. You can assume that the each word is separated by a white space or a new line.

Name of Python file: wordcounter.py

Total points: 10