Homework 4: Data of all types

Comp 123

Overview

Indivisual Work

This assignment should be done as individual work. Be sure the work is your own. Do not seek out solutions online. DO ask for help from me or our preceptors. DO write down any people or web sites where you get some measure of help.

Docstrings

For this assignment, I will be taking seriously the expectation that you include a docstring for every function. Leaving a docstring out for any function will cost one point per function. So long as your docstring reasonably describes the function you will be fine. For example, If I were writing a function to get the length of a cats tail:

good docstring (full points):

```
def getTailLength(cat):
    """Given a cat (the input parameter) this function computes the length
       of the cat's tail. The return of this function is the length of the
       cat's tail represented as an number, measured in inches."""
```

bad docstring (full points but you make me sad):

```
def getTailLength(cat):
    """it tells you hwo longg  a cats tail is"""
```

Assignment Goals

The goal of this assignment is to practice using the boolean data type, string data type, list data type and dictionary data type.

Preparing and handing in the assignment.

Download the `hw3code.py` and `hw3tests.py` files to use with this assignment. Put all answers into `hw3code.py`. Any non-code answers should be put in comments or a large, triple quoted string. Please do not re-name your copy of `hw3code.py`, If you do the test code will not work.

Assignment questions

1. (5 pts) Which of the following Python expressions evaluate to a Boolean value (with type `bool`)? Assume the variable definitions given here:

```
x = 25
y = 103
s = 'ha ha ha'
```

   (a) `'ha' in s`

   (b) `'True'`

   (c) `(x > y) and (y / x > 2)`

   (d) `3 or 4`

   (e) `not 'false'`

2. (10 pts) Define a function `sportsScore` the t̂sportsscore function takes two inputs, the first being a dictionary (with string keys and number values) and the second being a list or strings. The `sportsScore` function will be used for computing how many points a team scores in a sport. The list will describe what happens to the team in the game and the dictionary would describe how many points the action is worth. For an example, a soccer game might go `["kick off", "pass", "pass", "breakaway", "goal", "kick off", "pass", "pass", "goal"]` with the dictionary being `{"kick off":0, "pass":0, "breakaway":0, "goal":1}` leading to a score of 2 (the two goals one point each). Another example (american football) `["kick off", "first down", "field goal","kick off", "touchdown", "extra point"]` with the dictionary being `{"kick off": 0, "first down":0, "field goal":3, "touchdown":6, "extra point":1}` leading to a score of 10 points. The return value of `sportsScore` is the score computed by adding the scores of each action taken in the game. You may assume that the dictionary contains a score for every string in the list. For examples of this function being called and its expected outcome see the test code.

3. (10 pts) Define a function called `badWordFilter` that takes takes a string and a list of strings as its two inputs. The purpose of this function is to remove bad words from a piece of text. The string is the text to filter and the list of strings is the list of bad words. The function should return a modified version of the original string where every bad word has been replaced with the string `"<CENSORED>"`

   For examples of this function being called and its expected outcome see the test code.

   This function is a simplistic approach to censorship, do not expect it to work perfectly. For example, it is OK at this point if your censoring function is fooled by text which capitalizes words in a strange way to avoid the filter.

4. (10 pts) For this question, I have provided you with a program made up of two functions. The main function, `scrapeNames`, takes a string of text as its input. It breaks the text up into separate words, removing punctuation. It then searches through the list of words, looking for capitalized words. It wants to keep track of how many times each capitalized word occurs in the text. To store that information, it uses a dictionary, where the keys for the dictionary are the capitalized words, and the value associated with each key is the number of times that word has been seen.

The main function calls a helper function, `clean`, which takes a list of words and returns a new list of words, where the punctuation has been removed from the list.

This program has five bugs in it, spread across both functions. Read through the functions carefully, and then use the testing functions I have provided in the `hw4Tests.py` file to help you to find and correct the bugs. When you find a bug, put a comment on the line it is on, explaining what the bug was, and how you fixed it.

Hints:

- Debug the helper function, `clean`, first. Once it seems to work fine, then work on `scrapeNames`.
- Try walking through a call to one of the functions on paper, line by line, writing down the values of each variable.
- Remember the Wing debugger, and use it to step through the program if need be.
- Try putting print statements in to print out the status of the main variables at the start of a function call, and inside any loops or conditionals.