15-110: Principles of Computation
Lab 3: Sentence Analysis and GUIs
Released: Wednesday, 7-12
Due: Sunday, 7-17, 11:59 PM

Introduction:
Sometimes it isn't enough to just solve a problem. In the past, you have analyzed DNA strands and calculated odds. Unfortunately, the only people who can reap the benefits of your labor are other python programmers. If you want a lay person to use your programs, you must first create a GUI.

Background:

Entry():
This lab requires the use of the the `Entry` widget. The `Entry` widget is a small text box which can be written in by the user and changed by python. You can create a `Entry` widget named foo with the following syntax, with the "`. . .`" standing in for additional options:

```
foo = Entry(master, . . .)
```

Options:
Although there are many options that can be used in the creation of an `Entry` widget, only one will be used in this lab:

`width = foo`: The entry widget will be able to show foo characters at once, where `foo` is an integer.

Methods:
Once again, there are many methods that go along with the Entry widget. In this lab, we will only concern ourselves with the following three:

`foo.get()`: returns the string currently written in `foo`.

`foo.insert(index, text)`: writes the string text into `foo`, starting at index.

`foo.delete(start, end)`: deletes all text starting the index `start` and up-to, but not including the index `end`. To delete all text in `foo`, write `foo.delete(0, END)`. `END` is a command similar to `len(string)` for the purposes of this method.

Tkinter Problems:
Several people have reported various problems with Tkinter which are believed to be caused by differences in operating systems. If you believe that you are encountering such a problem or are incapable of using Tkinter on your computer, we suggest working on the cluster machines in GHC 3000.

Task 1: Sentence Analysis
For Task 1, you will write the functions that will be displayed on your GUI:

Task 1.1: Identifying Sentences
Write the function `isSentence(sentence)`. This function will take the string `sentence` as a parameter and return True if that string is a valid sentence. A string can be considered valid if

and only if:

-It contains at least one word.
-Words are never separated by more than one space.
-The first letter of the sentence is capital.
-At the end of the sentence, there is exactly one punctuation mark: an exclamation point, a question mark, or a period.

Task 1.2: Counting words

Write the function `countWords(sentence)`. This function will take the string `sentence` as a parameter and return an integer representing the number of words in that sentence. If the sting is not a sentence, `countWords` should return -1.

Task 1.3: Counting Unique Words

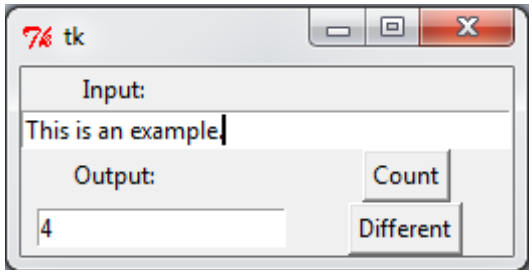
Write the function `differentWords(sentence)`. This function will take the string `sentence` as a parameter and return the number of unique word types (i.e. the number of different words) in that sentence. If the string is not a sentence, `differentWords` should return -1.

Task 2: Designing a GUI

It's all well and good to write a block of code that can count the number of words in a sentence, but you will never get rich and famous unless you can market that code to non-programers. If you want your name to spread across the world, you must make your code more accessible via a GUI.

Task 2.1: Placing the Widgets

Write the function `run()`, which will contain the required Tkinter set-up code and place your widgets in their correct places. Your GUI should consist of a `Frame` which contains two `Entry` widgets, two `Button` widgets, and two `Label` widgets. One entry field should be the input field where the user will be expected to type their text and the other should be the output field where your GUI will print out its answer. Both these fields should be labeled. In addition to these widgets, you should also have two buttons, labeled `Count` and `Different`. They should call the functions `count()` and `different()` when pressed.
Your layout must look like this:



Task 2.2: Connecting Buttons and Functions

Write the functions `count()` and `different()`.When the `Count` button is pressed, your GUI will count the number of words in the input box, writing the answer in the `Output` box. When the `Different` button is pressed, your GUI will count the number of unique words in the Input box, writing the answer in the `Output` box. All text that was inside the `Output` box should be deleted prior to writing the answer.

You will need to use global variables to pass the two `Entry` widgets into `count()` and different(), since the methods required to pass parameters through button commands require material that we have not covered yet.

If you find a mistake in this lab, please email **staff-110@cs.cmu.edu**.