CS1210 Lecture 18 Oct. 4, 2021

- Quiz 2 Wednesday, Oct. 6, in class
- HW4 due tomorrow
- There was some clear cheating on HW3 - uses of code that's been on the Internet for several years with idiosyncratic bug. So your initial score *might* not be your final score. I haven't addressed the Academic Honesty Policy violations yet …

Last time

- Tuples (10.26)
- Default and optional arguments to functions
- HW4
  - Sorting for HW4
- zip, generators and iterators

Today

- HW4 final questions/advice
- Simple image editing/manipulation
- Information about Quiz 2

HW4

It is interesting, and not hard if you do a little bit at a time. Get it working bit by bit.

1. Read the file, storing all the messages and their labels (spam/ham). E.g.
   - Two separate lists: ham list [['text', 'me', 'later!'], ['…', …], ...] and spam list [['call', '1412', 'to', 'win'], …] *(I recommend this option)*
   - Or one list [['spam', ['call', '1412', 'to', 'win']], ['ham', ['text', 'me', 'later!']], […], …]
   - Note: don't keep ham/spam label/tag as part of message. I've seen people do this and then write special case code to ignore 'ham'/'spam' when processing message words in step 2 below – this can yield errors.
2. Create a ham and a spam dictionary. For each message, extract its words, and update spam or ham dictionary of word counts accordingly
   - for 'text me later!' increment 'text', 'me', 'later' entries in ham dict
3. Use the two dictionaries to compute and print some statistics
   - get total spam/ham word counts and unique word counts
   - extract most common words from dictionaries
   - print stats

HW4

1. File has some non-Ascii characters.

   use: open(fileName, encoding = 'utf-8’)

2. To break line into tokens – individual elements of a line, learn how to use string **split**

   for line in fileStream:

   lineAsList = line.split() lec16split.py

3. get rid of extra stuff “…cool!?” learn how to use string **strip** (and/or lstrip, rstrip)

   – I *strongly recommend against* using replace() method
   – *Don’t put “” (empty string) as a word in your dictionaries*
   – *A line like*

     msgWord.strip(“!?,.;: “)

   does NOT work! Why?

for HW4, sorting is very helpful

Why?

You’ll have two dictionaries of the form:

{‘free’: 23, ‘you’: 50, ‘go’: 10, ‘zoo’: 1}

You’ll need to extract words in order from most to least frequent?

sort/sorted “work” on dictionaries but do they do what we want?

```
>>> d = {‘free’: 23, ‘you’: 50, ‘go’: 10, ‘zoo’: 1}
>>> sorted(d)
['free', 'go', 'you', 'zoo']
```

helpful???

For HW4, sorting is helpful

But suppose we have a list of tuples instead of a dictionary.

```
tl = [(‘free’, 23), (‘you’, 50), (‘go’, 10), (‘zoo’, 1)]
```

```
>>> sorted(tl)
[(‘free’, 23), (‘go’, 10), (‘you’, 50), (‘zoo’, 1)]
```

Now helpful? Not very.
sorts based on whole tuple

sorted (and sort) have two useful optional arguments:

key: you provide a little function that to apply to item to generate key to use to sort

reverse: provide True if you want list from largest to smallest instead of default of smallest to largest

For HW4, sorting is helpful

sorted (and sort) have two useful optional arguments:

1) key: a little function that is applied to item to generate key to use to sort

```
>>> tl = [('free', 23), ('you', 50), ('go', 10), ('zoo', 1)]
>>>sorted(tl, key = item1)
>>>sorted(tl, key = lambda item: item[1])
```

if function item1 exists

But don't need to write a separate function. 'lambda' allows you to define an (anonymous) function anywhere

```
[('zoo', 1), ('go', 10), ('free', 23), ('you', 50)]
```

yes - better!

So … now also use the other optional argument - reverse

2) reverse: True if you want list from largest to smallest instead of default of smallest to largest

```
>>> sorted(tl, key = lambda item: item[1], reverse = True)
[('you', 50), ('free', 23), ('go', 10), ('zoo', 1)]
```

That's what we want!

lec17.py

For HW4, sorting is very helpful

How do you use this stuff in HW4?

You will create two dictionaries of word counts - one for ham, and one for spam
And you'll want to extract items with highest counts.

1. Saw (three slides back) that

    sort(dict)

didn't quite give us what we needed

2. Saw (in last two slides) that we can usefully sort list of tuples

So … can you make a list of tuples [… (word, count) …] from a dictionary?

- use list(d.items())
- use list(zip(list(d.keys(), list(d.values())) what is zip? see Lec 17

It's perfectly fine to do things that way in HW4. It turns out you can also sort the dictionary directly using sorted and keywords key and reverse. Experiment and see if you can figure out how ..

Side topic:
Simple image editing with Python

Next Time: Quiz 2

- Four questions, 20 points
- Focus on
  - iteration with both **for** and **while**
  - **Lists**
  - **Dictionaries**