Compsci 101
List Comprehensions, Global, Parallel Lists

Susan Rodger
October 6, 2022

K is for ...

- **Kernel**
  - Core of the OS, Core for Machine Learning

- **Keyboard - QWERTY or DVORAK**
  - **DVORAK:**
    - Key and (Key, Value) pair
      - Heart of a dictionary

Tiffany Chen

- Duke BS - IDM CS/Biology
- Stanford PhD Biomedical Informatics (CS and Biomedicine)
- Was Director of Informatics, Cytobank
- Now Group Product Manager at Chan Zuckerberg Initiative

“Part of the advantage of being interdisciplinary is that you can see the big picture when no one else can, and you can communicate to everyone else what that big picture is”

Announcements

- Assign 2 – Turtles due tonight!
- APT-3 due next Thursday
- Assign 3 - Transform out today, due Thursday, Oct 20
  - Sakai quiz Assign3 – Due Tues, Oct 18 (no grace day)
- No lab Friday - Enjoy Fall Break!
- APT Quiz 1 is Oct 13-Oct 17
  - Take during this time
  - Two parts – each part has two APTs
  - Each part is timed
APT Quiz 1

- Is your own work!
  - No collaboration with others!
  - Use your notes, lecture notes, your code, textbook
  - DO NOT search for answers!
  - Do not talk to others about the quiz until grades are posted

- Post private questions on Ed Discussion
  - We are not on between 9pm and 9am!
  - We are not on all the time, especially weekends
  - Will try to answer questions between 9am – 9pm
    - About typos, cannot help you in solving APTs
  - See 101 APT page for tips on debugging APTs

APT Quiz 1 Oct 13-17

- Opens 10/13 Noon
- Closes at 11pm 10/17 – must finish all by this time
- There are two parts based on APTs 1-3
  - Each part has two APT problems
  - Each part is 2 hours – more if you get accommodations
  - Each part starts in Sakai under tests and quizzes
  - Sakai is a starting point with countdown timer that sends you to a new apt page just for each part
  - Could do each part on different day or same days

- Old APT Quiz so you can practice (not for credit) – on APT Page
APT Pancakes

• How do you solve this (or any) problem?
  • 7 Steps!

• Some APTs are hard problems to solve (step 1-4)
  • Translating to code easy

• Some APTs have easy-to-see algorithms (step 5)
  • Translating to code is hard

APT: Pancakes

Problem Statement

You're a short-order cook in a pancake restaurant, so you need to cook pancakes as fast as possible. You have one pan that can fit capacity pancakes at a time. Using this pan you must cook numCakes pancakes. Each pancake must be cooked for five minutes on each side, and once a pancake starts cooking on a side it has to cook for five minutes on that side. However, you can take a pancake out of the pan when you're ready to flip it after five minutes and put it back in the pan later to cook it on the other side.

Write the method, minutesNeeded, that returns the shortest time needed to cook numCakes pancakes in a pan that holds capacity pancakes at once. See the examples.

Examples

1. numCakes = 0
capacity = 4

Returns: 0

It takes no time to cook 0 pancakes.

2. numCakes = 2
capacity = 2

Returns: 10

You cook both pancakes on one side for five minutes, then flip them over and cook each on the other side for another five minutes.
Step 1: Solve an instance
Three pancakes in a two-cake pan

- First 5 minutes
  - 2 half cooking
  - 1 uncooked

- Second 5 minutes
  - 2 half cooking
  - 1 almost cooked

Step 1: Solve an instance

- What kind of instances? Simple cases that are quickly solved
  - What are these in Pancake problem?

- Don’t solve for N, solve for 5 (generalize is step 3)
  - What to do when there are two parameters?
    - Fix one, vary the other one
    - Helps identify cases

WOTO-1 Pancakes

How to teach pancake Flipping

- http://www.youtube.com/watch?v=W_gxLKSsSIE
  - For longer, more complex robotic tasks
    - http://www.youtube.com/watch?v=4usoE981e7I
Problem

- Given a file of words, which word occurs the most
- For each word count how many times it occurs
- Determine which word has the highest count

Parallel Lists

- We will use parallel lists to track data
  - Each word is stored in a list named `words`
  - Word’s count is stored in a list named `counts`
  - # occurrences of `words[k]` is in `counts[k]`

```
["apple", "fox", "vacuum", "lime"]
[  5,    2,   25,   15  ]
```

- What happens when we read a word?

```python
Read word "vacuum"?
```

```
["apple", "fox", "vacuum", "lime"]
[  5,    2,   26,   15  ]
```

- What happens when we read a word?

```python
Read word "cat"?
```
Calculate word most often in file

```python
def wordOccursTheMost(fname):
    f = open(fname)
    words = []
    counts = []
    for line in f:
        line = line.strip()  # remove newline
        data = line.split()
        for word in data:
            if word not in words:
                words.append(word)
                counts.append(1)
            else:  # update word
                pos = words.index(word)
                counts[pos] += 1
    f.close()
```

List Comprehension
Accumulator in one line

```python
def onlyPos(nums):
    ret = []
    for n in nums:
        if n > 0:
            ret.append(n)
    return ret
print(onlyPos([1, 2, 3, -1, -2, -3]))
```

List Comprehension Syntax

- V is any variable: all list elements in order
- V_EXP is any expression, often use V
List Comprehension Syntax

- if part optional - `BOOL_EXP` is a Boolean expression usually using `V`

```python
ret = []
for V in LIST:
    ret.append(V_EXP)
```

```python
ret = [V_EXP for V in LIST]
```

```python
ret = []
for V in LIST:
    if BOOL_EXP:
        ret.append(V_EXP)
```

```python
ret = [V_EXP for V in LIST if BOOL_EXP]
```

List Comprehension Examples

```python
print( [n*2 for n in range(6)] )
```

```python
print( [n for n in range(10) if n % 2 == 1] )
```

```python
print( [n/2 for n in range(10) if n % 2 == 0] )
```

```python
lst = ['banana', 'pineapple', 'apple']
print( [c for c in lst if 'n' in c] )
```

```python
WOTO-3 List Comprehension Examples
```
Assignment 3: Transform

- Reading and writing files
  - We've seen how to read, writing is similar
  - Open, read, and close
  - Open, write, and close - `.write(...)`

- Apply a function to every word in a file
  - Encrypt and decrypt
  - Respect lines, so resulting file has same structure

Encrypting and Decrypting

- We give you:
  - Transform.py
  - Vowelizer.py - Removes vowels, then re-vowelize

- You implement
  - Pig Latin
  - Caesar cipher

- Challenge: Shuffleizer

Concepts in Starter Code

- Global variables
  - Generally avoided, but very useful
  - Accessible in all module functions

- FileDialog and tkinter
  - API and libraries for building UI and UX

- Docstrings for understanding!

Transform – Remove Vowels

- First line of twain.txt:

```
1 The Notorious Jumping Frog of Calaveras County
```

- Run Transform.py on twain.txt
- Set as: `doTransform("-nvw", Vowelizer.encrypt)`
  ```python
  #doTransform("-rvw", Vowelizer.decrypt)
  ```

- Results in new file: twain-nvw.txt
- First line of twain-nvw.txt is:

```
1 Th Ntrs Jmpng Frg f Clvrs Cnty
```
Transform – Get vowels back?

- First line of `twain-nvw.txt`:

```
1 Th Ntrs Jmpng Frg f ClvrsCnty
```

- Run `Transform.py` on `twain-nvw.txt`

- Set as:

```
#doTransform("-nvw", Vowelizer.encrypt)
doTransform("-rvw", Vowelizer.decrypt)
```

- Results in new file: `twain-nvw-rvw.txt`

- First line of `twain-nvw-rvw.txt` is:

```
1 oath antares jumping fargo fe cleavers county
```

Transform – Vowels summary

- First line in `twain.txt`

```
1 The Notorious Jumping Frog of Calaveras County
```

- After removing vowels – “encrypt”

```
1 Th Ntrs Jmpng Frg f ClvrsCnty
```

- After trying to re-vowelize – “decrypt”

```
1 oath antares jumping fargo fe cleavers county
```