Nicholas Bowman, Sonja Johnson-Yu, Kylie Jue **Section #4**
CS106AP **July 17, 2019**

Section #4: Solutions

1. Introduction to String Parsing

```
def remove_front(s):
    if len(s) <= 2:
        return s
    else:
        return s[2:]

def x_to_end(s):
    if 'x' in s:
        x_index = s.find('x')
        return s[x_index:]
    else:
        return ""


def simile_builder(s):
    if ':' not in s:
        return 'Couldn't make a simile, sad times.'
    else:
        colon_index = s.find(':')
        first_term = s[:colon_index]
        second_term = s[colon_index + 1:]
        return 'a ' + first_term + ' is like a ' + second_term

def exclaim(s):
    mark = s.find('!')
    if mark == -1:
        return ''
    # scan left from the exclamation mark
    i = mark - 1
    while i >= 0 and s[i].isalpha():
        i -= 1
    word = s[i + 1: mark + 1]
    if len(word) >= 2:
        return word
    return ''

def vowels(s):
    colon = s.find(':')
    if colon == -1:
        return ''
    # scan right from the colon
```

```
    i = colon + 1
    while i < len(s) and s[i].lower() in 'aeiou':
        i += 1
    word = s[colon + 1:i]
    return word
```

2. Extracting Email Hostnames

This is one possible approach to decomposing this problem. Think about other ways to structure your solution!

```
def extract_hostname(line):
    at_index = line.find('@')
    if at_index == -1:
        return ''
    # scan forward till the end of the word or line
    i = at_index + 1
    while i < len(line) and line[i] != ' ':
        if line[i].isalpha() or line[i] =='.':
            i += 1
        else:
            break # end the loop immediately

    hostname = line[at_index + 1: i]

    if len(hostname) < 4 or not '.' in hostname:
        return ''

    return hostname

def extract_all_hostnames(filename):
    hostnames = []
    with open(filename, 'r') as f:
        for line in f:
            hostname = extract_hostname(line)
            if hostname != '' and not hostname in hostnames:
                hostnames.append(hostname)
    hostnames = sorted(hostnames)
    return hostnames
```

3. Practice with Dictionaries

```
def int_counts(ints):
    counts = {}
    for num in ints:
        if not num in counts:
            counts[num] = 0
```

```
        counts[num] += 1
    return counts

def mutual_friends(phonebook_one, phonebook_two):
    result = {}
    for name in phonebook_one:
        phone_num = phonebook_one[name]
        if name in phonebook_two and phonebook_two[name] == phone_num:
            result[name] = phone_num
    return result
```

4. Tracing Your Way Through The Office

```
$ python3 the-office.py 2 -flax
6
$ python3 the-office.py 5 -scott
1
$ python3 the-office.py 4
0
```

5. Nested List Functions

```
def threes(n):
    outer = []
    # first = first num of inner 1, 2, ... n
    for first in range(1, n + 1):
        inner = []
        for i in range(first, first + 3):
            inner.append(i)
        outer.append(inner)
    return outer

def countdown(n):
    outer = []
    # count = count of numbers in inner list
    for count in range(1, n + 1):
        inner = []
        for i in range(10, 10 - count, -1):
            inner.append(i)
        outer.append(inner)
    return outer
```

6. Nested Dictionary Functions

```
def first_list(strs):
    firsts = {}
    for s in strs:
```

```
        if len(s) >= 1:
            ch = s[0]
            if ch not in firsts:
                firsts[ch] = []
            firsts[ch].append(s)
    return firsts

def suffix_list(strs):
    suffixes = {}
    for s in strs:
        if len(s) >= 2:
            # use suffix as key
            suffix = s[len(s)-2:]
            if suffix not in suffixes:
                suffixes[suffix] = []
            suffixes[suffix].append(s)
    return suffixes

def reverse_dict(animal_dict):
    reversed_dict = {}
    for person in animal_dict:
        animal = animal_dict[person]

        if animal not in reversed_dict:
            reversed_dict[animal] = []

        reversed_dict[animal].append(person)

    for animal in sorted(reversed_dict.keys()):
        print(animal + ': ' + ', '.join(reversed_dict[animal]))
```

7. Big Tweet Data

```
def add_tweet(user_tags, tweet):
    user = parse_user(tweet)
    if user == '':
        return user_tags

    # if user is not in there, put them in with empty counts
    if user not in user_tags:
        user_tags[user] = {}

    # counts is the nested tag -> count dict
    # go through all the tags and modify it
    counts = user_tags[user]
    parsed_tags = parse_tags(tweet)
    for tag in parsed_tags:
        if tag not in counts:
            counts[tag] = 0
        counts[tag] += 1

    return user_tags

def parse_tweets(filename):
    user_tags = {}
    with open(filename, 'r') as f:
        for line in f:
            add_tweet(user_tags, line)
    return user_tags

def user_total(user_tags, user):
    """
    Optional. Given a user_tags dict and a user, figure out the total
count
    of all their tags and return that number.
    If the user is not in the user_tags, return 0.
    """
    if user not in user_tags:
        return 0
    counts = user_tags[user]
    total = 0
    for tag in counts.keys():
        total += counts[tag]
    return total
```