CompSci 101
Introduction to Computer Science

April 18, 2017
Prof. Rodger

Announcements
• No more RQ!
• Assign 8 due Thurs., Assign9 due Apr. 26
• APT 10 due Tues, Apr 25
• Lab this week
• Exam 2 back

• Today:
  – Review Recursion
  – Regular Expressions
  – Assignment 8 Recommender

Assignment 9 Due Apr 26
Shhh! No late penalty til Apr 28!
• Write a song, make a video about your experience with CompSci 101

Assignment 8
From User Rating to Recommendations

<table>
<thead>
<tr>
<th>Spectre</th>
<th>Martian</th>
<th>Southpaw</th>
<th>Everest</th>
<th>PitchPerfect 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>-3</td>
<td>5</td>
<td>-2</td>
<td>-3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>-2</td>
<td>1</td>
<td>-1</td>
</tr>
</tbody>
</table>

What should I choose to see?
  ➢ What does this depend on?
Who is most like me?
  ➢ How do we figure this out
ReadAllFood modules: Food Format

• All Reader modules return a tuple of strings: itemlist and dictratings dictionary

ReadFood first!

– Read input and save it

– Get list of restaurants – use that ordering! Set?

– For each person

  • For each restaurant and its rating
    – Must find location of restaurant in itemlist
    – Then update appropriate counter

– Print any structure you create to check it

Translated to:

```python
[['Kim Lee', [1, 0, 3, -3, -1, 5, 1, 0]], ('Charlotte', [3, 3, 5, 0, 0, 1, -3, -2]), ('Jack Jack', [-1, 1, 3, 1, -3, 5, 3]), ('Java', [3, 3, 5, 3, 1, 3, -1, 0]), ('Python', [1, 3, 5, 0, 3, 3, 0]), ('Xiaobai', [0, 5, 5, -5, 1, 0, -1, -3]), ('Jo Jo', [0, 0, 0, 1, 3, -1, 1])]
```

Follow 12-step process

Recursion Review

• Function calls a clone of itself
  – Smaller problem
  – Must be a way out of recursion

Example

```python
def Mystery(num):
    if num > 0:
        return 1 + Mystery(num/2)
    else:
        return 2 + num
```

• Mystery(5) is $1 + \text{Mystery}(2) = 1 + 4 = 5$
• Mystery(2) is $1 + \text{Mystery}(1) = 1 + 3 = 4$
• Mystery(1) is $1 + \text{Mystery}(0) = 1 + 2 = 3$
• Mystery(0) is $2$
Review: Recursion to find ALL files in a folder

- A folder can have sub folders and files
- A file cannot have sub files

```python
def visit(dirname):
    for inner in dirname:
        if isdir(inner):
            visit(inner)
        else:
            print name(inner), size(inner)
```

Something Recursion

```python
def Something(data):
    # data is a list of integers
    if len(data) == 0:
        return 0
    if data[0]%2 == 0:  # it is even
        return data[0] + Something(data[1:])
    else:
        return Something(data[1:])
```

Revisit the APT Bagels Recursively

```python
filename: Bagels.py

def bagelCount(orders):
    """
    return number of bagels needed to fulfill the orders in integer list parameter orders
    """
1. orders = [1,3,5,7]
   Returns: 16
   No order is for more than a dozen, return the total of all orders.
2. orders = [11,22,33,44,55]
   Returns: 175 since 11 + (22+1) + (33+2) + (44+3) + (55+4) = 175
```

APT Bagels Recursively

```python
A) def bagelCount(orders):
    if len(orders) > 0:
        return orders[0]/12 + orders[0] + bagelCount(orders[1:]):
    else:
        return 0
B) def bagelCount(orders):
    if len(orders) > 0:
        return orders[-1]/12 + orders[-1] + bagelCount(orders[:-1]):
    else:
        return 0
C) def bagelCount(orders):
    return orders[0] + orders[0]/12 + bagelCount(orders[1:]):
D) def bagelCount(orders):
    if len(orders)>1:
        return orders[1] + orders[1]/12 + bagelCount(orders[2:]):
    else:
        return bagelCount(orders[0])
```
Recursion in Pictures
• http://xkcd.com/543/

What is Computer Science?
• … "it is the study of automating algorithmic processes that scale."

• If you need to find one email address on a webpage, you don't need computer science
  – If you need to scrape every email address, that number in the 10's to 100's, you could use help

How do you solve a problem like …
• How many words end in "aria"?
  – Start with "aria"? Contain "aria"?
  – Why would you care about this?

• Can you find ola@cs.duke.edu, susan.rodger@duke.edu, and andrew.douglas.hilton@gmail.com when searching through a webpage source?
  – What is the format of a "real" email address?
Examples of regex's at work

- What do aria$ and ^aria and aria share?
  - Answers to previous question
- What about the regex .+@.+?
  - Turns out that . has special meaning in regex, so does +, so do many characters

- We'll use a module RegexDemo.py to check
  - Uses the re Python library
  - Details won't be tested, regex knowledge will

Regex expressions

- Regex parts combined in powerful ways
  - Each part of a regex "matches" text, can extract matches using programs and regex library
  - ^ is start of word/line, $ is end
- Expressions that match single characters:

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[a-zAB]</td>
<td>Brackets create character class</td>
</tr>
<tr>
<td>(regex)</td>
<td>Tag or group a regex</td>
</tr>
<tr>
<td>\1 or \2</td>
<td>Matches previously grouped regex</td>
</tr>
<tr>
<td>{1} or {n}</td>
<td>Repeat regex 1 or n times</td>
</tr>
</tbody>
</table>

Regex examples tried and explained

- Five letter words ending in p? Starts 'd'?
  - ^\w\w\w\wp$ but not ....p$
- Seven letter words, or seven ending with 'z'
  - Difference between ^\w{7}$ and ^\w{7}|
- Words that start with a consonant:
  - ^[^aeiou]$ double meaning of ^
Regex examples tried and explained

• Five letter words ending in p? Starts 'd'?
  \- `^\w\w\w\wp\$` but not `\p\$`

• Seven letter words, or seven ending with 'z'
  \- Difference between `^\w{7}\$` and `^\w{7}`

• Start and end with the same two letters like sense and metronome, decipher this:
  \- `^\(\w\w\)\.*\1\$`

• Start and end with three letters reversed, like despised and foolproof?

Summary of Regular Expressions

<table>
<thead>
<tr>
<th>regex</th>
<th>purpose</th>
<th>regex</th>
<th>purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>any character</td>
<td>*</td>
<td>zero or more of previous regex</td>
</tr>
<tr>
<td>\w</td>
<td>any alphanumeric character (and _)</td>
<td>+</td>
<td>one or more of previous regex</td>
</tr>
<tr>
<td>\s</td>
<td>any whitespace character</td>
<td>*? or +?</td>
<td>non-greedy version of either * or +</td>
</tr>
<tr>
<td>\d</td>
<td>any digit character</td>
<td>(</td>
<td>tag/group a regular expression</td>
</tr>
<tr>
<td>[]</td>
<td>character class, e.g., [A-Z] or [aeiou]</td>
<td>\1, \2, ..</td>
<td>match numbered flagged/grouped regex</td>
</tr>
<tr>
<td>^</td>
<td>beginning of line/string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[^...]</td>
<td>not the characters in the class, e.g., [^aeiou]</td>
<td>$</td>
<td>end of line/string</td>
</tr>
</tbody>
</table>

Regex Questions
bit.ly/101s17-0418-3

Answer Questions
bit.ly/101s17-0418-4

SortByFreqs APT
Sort items by their frequency, then sorted in frequencies.

data = ['"apple", "pear", "cherry", "apple", "pear", "apple", "banana"]

Returns: ['"apple", "pear", "banana", "cherry"]
Take Exam questions