CompSci 101
Introduction to Computer Science

Feb 24, 2015

Prof. Rodger
Announcements

• Reading and RQ11 due next time
• Assignment 5 out today - Hangman
• APT 4 due, APT 5 out
• Exam 1 back today
  – Exam regrades go to Prof. Rodger
  – Write on the front the problem and what the issue is
• Today
  – sets
  – Finish lecture notes from last time
More on List Comprehensions

• What is the list for the following:
  1) \([j+1 \text{ for } j \text{ in } \text{range}(20) \text{ if } (j\%3) == 0]\)
  2) \([i*2 \text{ for } i \text{ in } [j+1 \text{ for } j \text{ in } \text{range}(20) \text{ if } (j\%3) == 0] \text{ if } i*i > 19]\)
Problem: Given a list of strings, return the longest string. If there are more than one of that length, return the first such one.

['kiwi', 'plum', 'orange', 'lemon', 'banana']

Write a list comprehension for this problem.
Python Sets

• Set – unordered collection of distinct items
  – Unordered – can look at them one at a time, but cannot count on any order
  – Distinct - one copy of each

• Operations on sets:
  – Modify: add, clear, remove
  – Create a new set: difference(-), intersection(&), union (|), symmetric_difference(^)
  – Boolean: issubset <=, issuperset >=

• Can convert list to set, set to list
  – Great to get rid of duplicates in a list
Summary (from wikibooks)

- `set1 = set()`                   # A new empty set
- `set1.add("cat")`                # Add a single member
- `set1.update(["dog", "mouse"])`  # Add several members
- `set1.remove("cat")`             # Remove a member - error if not there
- `print set1`                     # Iteration or “for each element”
- `print "Item count:", len(set1)`  # Length, size, item count
- `isempty = len(set1) == 0`        # Test for emptiness
- `set1 = set(["cat", "dog"])`    # Initialize set from a list
- `set3 = set1 & set2`              # Intersection
- `set4 = set1 | set2`              # Union
- `set5 = set1 - set3`              # Set difference
- `set6 = set1 ^ set2`              # Symmetric difference (elements in either set but not both)
- `issubset = set1 <= set2`         # Subset test
- `issuperset = set1 >= set2`       # Superset test
- `set7 = set1.copy()`              # A shallow copy (copies the set, not the elements)
- `set8.clear()`                   # Clear, empty, erase
Creating and changing a set

colorList = ['red', 'blue', 'red', 'red', 'red', 'green']
colorSet = set(colorList)
smallList = list(colorSet)
colorSet.clear()
colorSet.add("yellow")
colorSet.add("red")
colorSet.add("blue")
colorSet.add("yellow")
colorSet.add("purple")
colorSet.remove("yellow")

• See setsEasy.py
Set Operations

```python
UScolors = set(["red", "white", "blue"])  
dukeColors = set(["blue", "white"])  
print dukeColors.union(UScolors)  
print dukeColors | UScolors  
print dukeColors.intersection(UScolors)  
print dukeColors & UScolors  
print dukeColors.difference(UScolors)  
print dukeColors - UScolors  
print UScolors - dukeColors  
print dukeColors ^ UScolors  
print UScolors ^ dukeColors

• See setsEasy.py
```
poloClub = set(['Mary', 'Laura', 'Dell'])
rugbyClub = set(['Fred', 'Sue', 'Mary'])

Question 1:
print [w for w in poloClub.intersection(rugbyClub)]

Question 2:
print [w for w in poloClub.union(rugbyClub)]
More Set Examples

bit.ly/101S15-0224-03

lista = ['apple', 'pear', 'fig', 'orange', 'strawberry']
listb = ['pear', 'lemon', 'grapefruit', 'orange']
listc = [x for x in lista if x in listb]
listd = list(set(lista) | set(listb))

Question 1:
print listc

Question 2:
print listd