Announcements

- Reading and RQ due next time
- Assignment 5 out today
- APT 4 due today, APT 5 out
- Lab 6 this week
  - Read APT Anagramfree and Assignment 5 before going to lab!
- Today:
  - Focus on problem solving with sets, list comprehensions

Duke ACM

- ACM – Computer Science student club
  - Association of Computing Machinery
- Event tonight! 6pm Physics 128
- Tech Entrepreneurship talk with Envested CEO and CTO
  bit.ly/dukeacmtechtalk
- Join the dukelist for dukeacm

Grace Hopper Celebration of Women in Computing

- Apply for fellowships to go to the conference, by march 30!
  - Pays your way to the conference
- 15,000 women in computing
  ghc.anitaborg.org
More on List Comprehensions

Problem: Given a list of strings, return the longest string. If there are more than one of that length, return the first such one.


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
- for item in set1:
  - print item
- 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
- set2 = 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", "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")

Set Operations

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
Set Examples

```python
poloClub = set(['Mary', 'Laura', 'Dell'])
rugbyClub = set(['Fred', 'Sue', 'Mary'])
```

Questions:
```python
print [w for w in poloClub.intersection(rugbyClub)]
print poloClub.intersection(rugbyClub)
print [w for w in poloClub.union(rugbyClub)]
print poloClub.union(rugbyClub)
```

Set Examples (cont)

```python
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))
```

Assignment 5 - Hangman

- Guess a word given the number of letters.
  - Guess a letter
  - see if it is in the word and where.

- Demo

- Will start in lab

APT AnagramFree

```python
words = ['creation', 'sentence', 'reaction', 'sneak', 'star', 'rats', 'snake']
```

Returns: 4

- "star" "rats" → both have letters: a r t s
- "snake" "sneak"
- "creation" "reaction"
- "sentence"
Problem

- Given two books:
  - How many words in each book?
  - How many unique words in each book?
  - What words that start with “r” are in one book and not the other book?