M is for ...

- **Machine Learning**
  - Math, Stats, CompSci: learning at scale
- **Microsoft, Mozilla, Macintosh**
  - Software that changed the world?
- **Memory**
  - Storage space in the computer
  - From 64 Kilobytes to 16 Gigabytes!
- **Mouse, Mouse pad**
  - Easier to navigate
Margot Shetterly

- Writer, Author of Hidden Figures
- Black Women NASA Scientists
- Gave a talk at Duke in 2016

Katherine Johnson, Mary Jackson, Dorothy Vaughn, Christine Darden
Announcements

• Assignment 3 due Thursday, March 2
  • Sakai quiz due today

• Assignment 4 out Thursday!

• APT-4 is out and due Thursday March 9
  • Can use some as practice for exam

• Lab 7 Friday, there is a prelab available Thursday!

• Do not discuss APT Quiz 1 until grades posted!
  • A few have not take it yet due to travel or illness
PFTD

• Simple Sorting
• Solving an APT
• Assignment 4
• Sets
Exam 2 – in person – Tues, March 7

- Exam is in class on paper – 10:15am
  - Need pen or pencil
- See materials under 3/7 date
  - Exam 2 Reference sheet - part of exam
- Covers
  - topics /reading through Thursday
  - APTs through APT4
    - APT4 – write code on paper, then type in
  - Labs through Lab 7
    - Lab 7 - Parts 1-3
  - Assignments through Assignment 3
Exam 2 topics include ...

• List, tuples, list comprehensions
• Loops – for loop, while loop, indexing with a loop
• Reading from a file
  • Converting data into a list of things
• Parallel lists
• Sets – solving problems
• Dictionaries – only reading them and understanding output, no problem solving
• No turtles on the exam!
Exam 2

• Exam 2 is your own work!
• No looking at others exam or talking to others
• You cannot use any notes, books, computing devices, calculators, or any extra paper
• Bring only a pen or pencil
• The exam has extra white space and has the Exam 2 reference sheet as part of the exam.

• Do not discuss any problems on the exam with others until it is handed back
Exam 2 – How to Study

• Practice writing code on paper!
• Rewrite an APT
• Try to write code from lecture from scratch
• Try to write code from lab from scratch
• Practice from old exams
• Put up old Sakai quizzes, but better to practice writing code
• Look at Exam 2 reference sheet when writing code!
Let’s sort lists with sorted() function

- **Want list elements in sorted order**
  - Example: have list [17, 7, 13, 3]
  - Want list [3, 7, 13, 17], in order

- **Built-in function: sorted(sequence)**
  - **Returns new list** of sequence in sorted order
  - Sequence could be list, tuple, string
Example

```python
lst = [6, 2, 9, 4, 3]
lsta = sorted(lst)
b = ['ko', 'et', 'at', 'if']
c = sorted(b)
b.remove('et')
b.append(6)
b.insert(1,5)
c = sorted(b)
```

```
lst is [6, 2, 9, 4, 3]
lsta is [2, 3, 4, 6, 9]
c is ['at', 'et', 'if', 'ko']
b is ['ko', 'at', 'if']
b is ['ko', 'at', 'if', 6]
b is ['ko', 5, 'at', 'if', 6]
ERROR!!!!!!!!! Cannot sort
numbers and strings
```
Example

lst = (7, 4, 1, 8, 3, 2)
lsta = sorted(lst)
b = ('ko', 'et', 'at', 'if')
c = sorted(b)
d = "word"
e = sorted(d)
f = 'go far'
g = sorted(f)
h = sorted(f.split())
Now, sort lists with `.sort()` list method

- Want to "change" list elements to sorted order
  - lst is [17, 7, 13, 3]
  - lst.sort()
  - Now **same** list lst is [3, 7, 13, 17], in order

- **List method**: `list.sort()`
  - List is **modified, now in sorted order**
  - There is NO return value
  - Only works with lists, can’t modify strings, tuples
Compare sorted() with .sort()

lsta = [6, 2, 9, 4, 3]
lstb = sorted(lsta)

lsta.sort()
a = [7, 2, 9, 1]
b = a.sort()

c = (5, 6, 2, 1)
c.sort()
d = "word"
d.sort()
Problem Statement

Strange abbreviations are often used to write text messages on uncomfortable mobile devices. One particular strategy for encoding texts composed of alphabetic characters and spaces is the following:

- Spaces are maintained, and each word is encoded individually. A word is a consecutive string of alphabetic characters.
- If the word is composed only of vowels, it is written exactly as in the original message.
- If the word has at least one consonant, write only the consonants that do not have another consonant immediately before them. Do not write any vowels.
- The letters considered vowels in these rules are 'a', 'e', 'i', 'o' and 'u'. All other letters are considered consonants.

For instance, "ps i love u" would be abbreviated as "p i lv u" while "please please me" would be abbreviated as "ps ps m". You will be given the original message in the string parameter original. Return a string with the message abbreviated using the described strategy.

Specification

```python
filename: TxMsg.py
def getMessage(original):
    ""
    return String that is 'textized' version of String parameter original ""
    # you write code here
```
Examples

1. "text message"
   Returns "tx msg"

5. "aeiou bcdfghijklmnpqrstuvwxyz"
   Returns: "aeiou b"
Write helper function \textit{transform}

- How?
- Use seven steps
- Work an example by hand
Why use helper function 'transform'?

- **Structure of code is easier to reason about**
  - Harder to develop this way at the beginning
  - Similar to accumulate loop, build on what we know

- **We can debug pieces independently**
  - What if transform returns "" for every string?
  - Can we test transform independently of getMessage?
Assignment 4: Guess Word

• We give you most of the functions to implement
  • Partially for testing, partially for guiding you
• But still more open ended than prior assignments
• If the doc does not tell you what to do:
  • Your chance to decide on your own!
    • Okay to get it wrong on the first try
  • Discuss with TAs and friends, brainstorm!
• Demo!
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

\[
x = [5, 3, 4, 3, 5, 1] \\
y = \text{set}(x)
\]

\[
x \text{ is } [5, 3, 4, 3, 5, 1] \\
y.\text{add}(6) \\
y.\text{add}(4)
\]
List vs Set

• List
  • Ordered, 3\textsuperscript{rd} item, can have duplicates
  • Example: \texttt{x = [4, 6, 2, 4, 5, 2, 4]}

• Set
  • No duplicates, no ordering
  • Example: \texttt{y = set(x)}

• Both
  • Add, remove elements
  • Iterate over all elements
Python Sets

• Can convert list to set, set to list
  • Great to get rid of duplicates in a list

\[
a = [2, 3, 6, 3, 2, 7] \quad \text{a is} \quad [2, 3, 6, 3, 2, 7]
b = \text{set}(a)
\]

\[
c = \text{list}(b)
\]
Python Set Operators

• Using sets and set operations often useful
• A | B, set union
  • Everything
• A & B, set intersection
  • Only in both
• B – A, set difference
  • In B and not A
• A ^ B, symmetric diff
  • Only in A or only in B
# List and Set, Similarities/Differences

<table>
<thead>
<tr>
<th></th>
<th>Function for List</th>
<th>Function for Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adding element</td>
<td><code>x.append(elt)</code></td>
<td><code>x.add(elt)</code></td>
</tr>
<tr>
<td>Size of collection</td>
<td><code>len(x)</code></td>
<td><code>len(x)</code></td>
</tr>
<tr>
<td>Combine collections</td>
<td><code>x + y</code></td>
<td>`x</td>
</tr>
<tr>
<td>Iterate over</td>
<td><code>for elt in x:</code></td>
<td><code>for elt in x:</code></td>
</tr>
<tr>
<td>Element membership</td>
<td><code>elt in x</code></td>
<td><code>elt in x</code></td>
</tr>
<tr>
<td>Index of an element</td>
<td><code>x.index(elt)</code></td>
<td>CANNOT DO THIS</td>
</tr>
</tbody>
</table>

- Lists are ordered and indexed, e.g., has a first or last
- Sets are **not** ordered, very fast, e.g., `if elt in x`
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")

smallList is
Set Operations – Union and Intersection

UScolors = set(['red', 'white', 'blue'])
dukeColors = set(['blue', 'white', 'black'])

print(dukeColors | UScolors)
print(dukeColors & UScolors)
Set Operations - Difference

UScolors = set(["red", "white", "blue"])
dukeColors = set(["blue", "white", "black"])

print(dukeColors - UScolors)
print(UScolors - dukeColors)
Set Operations – Symmetric Difference

UScolors = set(['red', 'white', 'blue'])
dukeColors = set(['blue', 'white', 'black'])

print(dukeColors ^ UScolors)
print(UScolors ^ dukeColors)
Let’s sort lists with sorted() function

• **Built-in function: sorted**(sequence)
  • **Returns new list** of sequence in sorted order
  • Sequence could be list, tuple, string
  • **Sequence could be set**!

\[
a = \text{set( [3, 5, 2, 1, 7, 2, 5] )}
\]
\[
b = \text{sorted}(a)
\]
WOTO-3 Sets