Compsci 101
Simple Sorting, Transform, Sets

Susan Rodger
Oct 18, 2022

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  Mary  Dorothy  Christine
Johnson  Jackson  Vaughn  Darden

Announcements

- APT-4 is out and due Thursday October 27
- Assignment 3 due Thursday, Oct 20
  - Sakai quiz due today
- Lab 6 Friday, there is a prelab available now!
- Do not discuss APT Quiz 1 until grades posted!
- All Assign, APT, APT quiz 2 dates now on calendar!
- Last chance for regrades for Exam 1 is tonight 11pm
Prof Rodger no office hours today

- I will be at the majors fair in Penn Pavilion
  - from 1-4pm if you want to ask questions about CompSci major

- Thursday office hours will be online only as I am traveling after class

PFTD

- Simple Sorting
- Solving an APT
- Sets

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)
```
Example

```python
lst = (7, 4, 1, 8, 3, 2)

lst = sorted(lst)
```

```python
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

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

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

Examples

1. "text message"
   Returns "tx msg"

5. "aeiou bcdfghjklmnpqrstuvwxyz"
   Returns: "aeiou b"

Write helper function `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?

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

```python
x = [5, 3, 4, 3, 5, 1]
y = set(x)
y.add(6)
y.add(4)
```

List vs Set

- **List**
  - Ordered, 3rd item, can have duplicates
  - Example: `x = [4, 6, 2, 4, 5, 2, 4]`

- **Set**
  - No duplicates, no ordering
  - Example: `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

```python
a = [2, 3, 6, 3, 2, 7]
b = set(a)
c = list(b)
```
Python Sets

- Operations on sets:
  - Modify:
    - add: a.add(7)
    - clear: a.clear()
    - remove: a.remove(5)
  - Create a new set: a = set([])

- difference(-), intersection(&), union (|), symmetric_difference(^)
- Boolean: issubset <=, issuperset >=

Python Set Operators

- Using sets and set operations often useful
- A | B, set union
  - Everything
- A & B, set intersection
  - Only in both
- A – B, 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>Function for List</th>
<th>Function for Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adding element</td>
<td>x.append(elt)</td>
</tr>
<tr>
<td>Size of collection</td>
<td>len(x)</td>
</tr>
<tr>
<td>Combine collections</td>
<td>x + y</td>
</tr>
<tr>
<td>Iterate over</td>
<td>for elt in x:</td>
</tr>
<tr>
<td>Element membership</td>
<td>elt in x</td>
</tr>
<tr>
<td>Index of an element</td>
<td>x.index(elt)</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.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)

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