Creating a list

- Given a list of numbers, create a second list of every number squared.

```python
nums = [8, 3, 5, 4, 1]
sqnums = []
for v in nums:
    sqnums.append(v*v)
print sqnums
```

```
[64, 9, 25, 16, 1]
```
List operations from book

<table>
<thead>
<tr>
<th>Method</th>
<th>Parameters</th>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>append</td>
<td>item</td>
<td>mutator</td>
<td>Adds a new item to the end of a list</td>
</tr>
<tr>
<td>insert</td>
<td>position, item</td>
<td>mutator</td>
<td>Inserts a new item at the position given</td>
</tr>
<tr>
<td>pop</td>
<td>none</td>
<td>hybrid</td>
<td>Removes and returns the last item</td>
</tr>
<tr>
<td>pop</td>
<td>position</td>
<td>hybrid</td>
<td>Removes and returns the item at position</td>
</tr>
<tr>
<td>sort</td>
<td>none</td>
<td>mutator</td>
<td>Modifies a list to be sorted</td>
</tr>
<tr>
<td>reverse</td>
<td>none</td>
<td>mutator</td>
<td>Modifies a list to be in reverse order</td>
</tr>
<tr>
<td>index</td>
<td>item</td>
<td>return idx</td>
<td>Returns the position of first occurrence of item</td>
</tr>
<tr>
<td>count</td>
<td>item</td>
<td>return ct</td>
<td>Returns the number of occurrences of item</td>
</tr>
<tr>
<td>remove</td>
<td>item</td>
<td>mutator</td>
<td>Removes the first occurrence of item</td>
</tr>
</tbody>
</table>

Problem

- Remove all negative numbers from list 
  \[4, -2, 5, 6, -3] \rightarrow [4, 5, 6]

- Two ways
  1) return a new list with all negative numbers removed
  2) Modify a list to remove negative numbers

www.bit.ly/101sp16-0225-1

```python
def removeNegatives(numberlist):
    answer = []
    for num in numberlist:
        if num >= 0:
            answer.append(num)
    return answer
```

somenums = [3, -1, 8, -5, -2, 6, 7]
nonegs = removeNegatives(somenums)


```python
def removeNegatives2(numberlist):
    for x in range(len(numberlist)):
        value = numberlist[x]
        if value < 0:
            numberlist.pop(x)
```

somenums = [3, -1, 8, -5, -2, 6, 7]
removeNegatives2(somenums)
def removeNegatives3(numberlist):
    pos = 0;
    while (True):
        if pos >= len(numberlist):
            break
        value = numberlist[pos]
        if value < 0:
            numberlist.pop(pos)
        pos = pos + 1

somenums = [3, -1, 8, -5, -2, 6, 7]
removeNegatives3(somenums)

List Comprehension
• Take advantage of patterns, make a new list
  based on per element calculations of another list

• Format:
  \[<expression with variable> for <variable> in
   <old list>\]

• Example:
  \[\text{nums} = [8, 3, 5, 4, 1]\]
  \[\text{sqnums} = [v^2 \text{ for } v \text{ in nums}]\]

Richard Stallman
• MacArthur Fellowship (Genious grant)
• ACM Grace Murray Hopper award
• Started GNU – Free Software Foundation (1983)
  – GNU Compiler Collection
  – GNU Emacs

These result in the same list!
\[
\text{nums} = [8, 3, 5, 4, 1]
\]

1) \[
\text{sqnums} = []
\]
   \[
\text{sqnums.append}(v^2)
\]

2) \[
\text{sqnums} = [v^2 \text{ for } v \text{ in nums}]
\]
Examples of List Comprehensions

nums = [4, 3, 8]
[v for v in nums]
[2 for v in nums]
sum([v*2 for v in nums])
[v+5 for v in nums][1]

Creating a list with just the even numbers

nums = [8, 3, 5, 4, 1]
evennums = []
for v in nums:
    if v % 2 == 0:
        evennums.append(v)
print evennums

[8, 4]

List Comprehension with Filtering

• Create list and use “if” to filter out elements to the list
• Format:
  • [<expression with variable> for <variable> in <old list> if <filter with variable> ]
• Example: nums = [8, 3, 5, 4, 1]
evennums =
  [v for v in nums if v%2==0]

More on List Comprehensions

names = [“Bo”, “Moe”, “Mary”, “Aaron”, “Joe”]
• What is the list for the following:
  1) [w for w in names if w.endswith(“e”)]
  2) [w for w in names if w.lower()[0] > ‘c’]
  3) [j+1 for j in range(20) if (j%3) == 0]
  4) [i*2 for i in [j+1 for j in range(20) if (j%3) == 0] if i*i > 19]
Giving Back Exam 1…

• Will post solutions
• Try working problem you missed first
  – Then look at solution

• Once you think you understand
  – Get blank sheet of paper – try again
• Understand all solutions

Exam 1 scores

84 84 84 84 84 84 83 83 83 83 83 83 83 83 83 83 83 83 83 83 83 83 83 83 83 82 82 82 82 82 82 82 82 82 82 82 82 82 82 82 82 82 82 ...
61 60 60 60 60 60 60 60 59 58 57 56 55 55 55 55 54 53 53 53 50 49 48 44 44 44 42 41 41 41 40 39 39 39 38 36 30 29 29 15