Announcements, Reminders, ...

- Any problems with CodeLab?
- Talk by Wendell Wallach on "Robot Minds and Human Ethics". Next Tuesday at 5:30pm in Social Sciences 016. (Cookies and Drinks at 5pm.)
- The ELIZA assignment is due next Tuesday.
- The reading for next Tuesday is on Blackboard.
- The solutions to the exam problems are also on Blackboard.

Warm-up Exercise

Write a function that reads in numbers from the user until the user enters *done* and then returns the smallest of the numbers that the user has entered.

What about this?

Write a function that reads in numbers from the user until the user enters *done* and then returns the ***median*** of the numbers that the user has entered.

Lists

- are a type of object for storing collections of other objects.
- are a built-in type in Python.

Some Python Lists

["apple", "banana", "pear"]

[1, 3, 5, -2, 97, 3]

[2, 7.3, "apple", 4]

[[1, 2, 3], "apple", 5]

[]

[["apple", [2, 4]], "blue", []]

Things to do with Lists in Python

Operators:

concatenation: +

repetition: *

indexing: [<*int*>]

slicing: [<*int*>:<*int*>]

Function:

length: len(<*list*>)

Lists and Strings

- are both sequences

but:

- Strings are **immutable**

  they *cannot be changed* once they have been created

- Lists are **mutable**

  they *can be changed* after they have been created

Variables and Mutable Objects

```
x = [1, 2, 3]
y  = x
y[3] = 4
print x
print y
```

Traversing Lists

Describe an algorithm that takes a list (of numbers) and returns the smallest number from the list.

Traversing Lists

Describe an algorithm that takes a list (of numbers) and returns the smallest number from the list.

```
def findmin (list):
   m = list[0]
   pos = 1
   while pos <
len(list):
      if list[pos] < m:
         m = list[pos]
      pos = pos + 1
   return m
```

Traversing Lists

Describe an algorithm that takes a list (of numbers) and returns the smallest number from the list.

```
def findmin (list):
   m = list[0]
   pos = 1
   while pos <
len(list):
      if list[pos] < m:
         m = list[pos]
      pos = pos + 1
   return m
```

```
def findmin2 (list):
   m = list[0]
   for item in list:
      if item < m:
         m = item
   return m
```

for-loops

Template:

```
for <var> in <sequence>:
   <statements>
```

Example:

```
def findmin2 (list):
   m = list[0]
   for item in list:
      if item < m:
         m = item
   return m
```

Practice Using for-loops

- Write a function `mylen` that takes a list as parameter and returns the length of the list. Do not use the built-in function len.
- Write a function `mycount` that takes a list and an expression as parameters and counts how many times the expression occurs in the list. For example:

```
>>> mycount([1,2,3,4,3,5],3)
2
>>> mycount(["a","a","a"],"b")
0
```

- Write a function `mymember` that takes a list and an expression as parameters and returns True if the expression occurs in the list and False if not.
- Write a function `myreverse` that takes a list as parameter and reverses it. For example:

```
>>> myreverse([1,2,3])
[3,2,1]
```