

# **Practice from Last Time**

Get the file Nov1.py from my Box.com code directory. It has the main function written for you and stubs for 2 other functions that you will need to write.

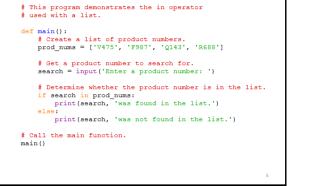
findAverage(numbers) – will return the average of all the numbers in the list

countNumbers(numbers, average) - will return 2 values; it counts the number of above average and below average numbers in a list

# Finding Items in Lists with the in Operator

- You can use the in operator to determine whether an item is contained in a list
  - General format: item in list
  - Returns  ${\tt True}$  if the item is in the list, or  ${\tt False}$  if it is not in the list
- Similarly you can use the **not in** operator to determine whether an item is not in a list

## Example Using in Operator



### List Methods and Useful Built-in Functions

- **append (item)**: used to add items to a list item is appended to the end of the existing list
- <u>index (item)</u>: used to determine where an item is located in a list
  - Returns the index of the first element in the list containing item
  - Raises ValueError exception if *item* not in the list

# find() doesn't exist for lists

list\_var.index(item)

- Searches left to right, returns position where found, but crashes if not found.
- Let's build an algorithm that replicates find(), but works for lists (returns -1 if not found).

# **Example Using Append**

def main():

```
infile = open("randomNums.txt", 'r')
numbers = []
for line in infile:
    numbers.append(int(line))
print(numbers)
```

main()

#### Output [62, 57, 55, 27, 45, 44, 46, 68, 86, 27, 88, 33, 11, 61, 64, 45, 56, 9, 33, 32, 56, 63, 24, 26, 100, 95, 62, 10, 87, 58, 69, 54, 75, 41, 22, 93, 82, 16, 92, 49, 6, 71, 85, 59, 56, 22, 3, 50, 1, 20, 54, 18, 27, 78, 17, 7, 41, 83, 92, 38, 5, 64, 60, 92, 15, 26, 57, 39, 0, 41, 67, 56, 24, 77, 28, 90, 24, 72, 2, 46, 75, 53, 58, 47, 50, 18, 40, 65, 24, 58, 41, 58, 81, 40, 6, 77, 85, 86, 68, 63]

8

### List Methods and Useful Built-in Functions (cont'd.)

- <u>insert(index, item)</u>: used to insert *item* at position *index* in the list
- <u>sort()</u>: used to sort the elements of the list in ascending order
- <u>remove (item)</u>: removes the first occurrence of *item* in the list
- **<u>reverse()</u>**: reverses the order of the elements in the list

## List Methods and Useful Built-in Functions (cont'd.)

- <u>del statement</u>: removes an element from a specific index in a list
  - General format: del list[i]
- <u>min and max functions</u>: built-in functions that returns the item that has the lowest or highest value in a sequence
  - The sequence is passed as an argument
- <u>sum function</u>: built-in functions that returns the total of all the values in a sequence

11

The sequence is passed as an argument

# Example Using del, min, max, and sum functions

my\_list = [5, 4, 3, 2, 50, 40, 30]
print("Before Deletion:", my\_list)
del my\_list[2]
print("After Deletion:", my\_list)

print("The lowest value is", min(my\_list))
print("The highest value is", max(my\_list))
print("The sum of values in my list is", sum(my\_list))

alpha\_list = ['a','b','c','d']
print("The lowest value is", min(alpha\_list))
print("The highest value is", max(alpha\_list))
# You cannot take the sum of a list that has strings in it
Output

Output	
Before Deletion: [5, 4, 3, 2, 50, 40, 30]	
After Deletion: [5, 4, 2, 50, 40, 30]	
The lowest value is 2	
The highest value is 50	
The sum of values in my list is 131	
The lowest value is a	
The highest value is d	

## Practice

Write a program that randomly generates 20 integers between 1 and 50, and stores them in a list. Print out the **lowest** and the **highest** numbers in your list, as well as the **sum** of all the numbers in the list.

Write a function that prints out sums of adjacent pairs of numbers in the list

 $\ensuremath{\mbox{Hint:}}$  You don't need the sliding window technique; instead, use math with list indices.

Write a function that takes a list and shifts all the elements in the list one spot to the left, without using slices! (the left-most element disappears) Example: [1, 2, 3, 4, 5] turns into [2, 3, 4, 5, 5]

13