Introduction to Problem Solving in Python

COSI 10A

Class objectives

- List-Traversal (Section 7.2)

Lists

List declaration

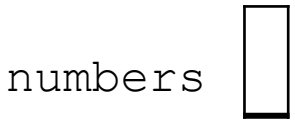
Syntax:

```
name = []
```

Creates an empty list

Example:

```
numbers = []
```



List functions

Function	Description
`append(x)`	Add an item to the end of the list. Equivalent to `a[len(a):] = [x]`.
`extend(L)`	Extend the list by appending all the items in the given list. Equivalent to `a[len(a):] = L`
`insert(i, x)`	Inserts an item at a given position. i is the index of the element before which to insert, so `a.insert(0, x)` inserts at the front of the list.
`remove(x)`	Removes the first item from the list whose value is *x*. Errs if there is no such item.
`pop(i)`	Removes the item at the given position in the list, and returns it. `a.pop()` removes and returns the last item in the list.
`clear()`	Remove all items from the list.
`index(x)`	Returns the index in the list of the first item whose value is *x*. Errs if there is no such item.
`count(x)`	Returns the number of times *x* appears in the list.
`sort()`	Sort the items of the list
`reverse()`	Reverses the elements of the list
`copy()`	Return a copy of the list.

Weather question 2

- Modify the weather program to print the following output:

```
Type in a temperature or "done" to finish
Day 1's high temp: 45
Day 2's high temp: 44
Day 3's high temp: 39
Day 4's high temp: 48
Day 5's high temp: 37
Day 6's high temp: 46
Day 7's high temp: 53
Day 8's high temp: done
Average temp = 44.6
4 days were above average.
```

Weather question 2 answer

```
def weather():
    print("Type in a temperature or \"done\" to finish")

    temps = []    # list to store days' temperatures
    sum = 0
    done = input("Day 1's high temp: ")
    day = 0

    while(done != "done"):          # read/store each day's temperature
        done = int(done)
        sum += done
        temps.append(done)
        day += 1
        done = input(("Day " + str(day + 1) + "'s high temp: "))

    average = sum / day

    count = 0     # see if each day is above average
    for i in range(0, day):
        if (temps[i] > average):
            count += 1
    # report results
    print("Average temp = " + str(average))
    print(str(count) + " days above average")
```

Weather question 3

- Modify the weather program to print the following output:

```
How many days' temperatures? 7
Day 1's high temp: 45
Day 2's high temp: 44
Day 3's high temp: 39
Day 4's high temp: 48
Day 5's high temp: 37
Day 6's high temp: 46
Day 7's high temp: 53
Average temp = 44.6
4 days were above average.

Temperatures: [45, 44, 39, 48, 37, 46, 53]
Two coldest days: 37, 39
Two hottest days: 53, 48
```

Weather question 3 answer

```
# Reads temperatures from the user, computes average and # days above average.
def main():
    days = int(input("How many days' temperatures? "))

    temps = [0] * days    # list to store days' temperatures
    sum = 0

    for i in range(0, days):       # read/store each day's temperature
        temps[i] = int(input(("Day " + (i + 1) + "'s high temp: ")))
        sum += temps[i]
    average = sum / days

    count = 0     # see if each day is above average
    for i in range(0, days):
        if (temps[i] > average):
            count += 1
    # report results
    print("Average temp = " + str(average))
    print(str(count) + " days above average")

    print("Temperatures: " + str(temps))
    temps.sort()
    print("Two coldest days: " + str(temps[0]) + ", " + str(temps[1]))
    print("Two hottest days: " + str(temps[-1]) + ", " + str(temps[-2]))
```

List mystery problem

- What element values are stored in the following list?

```
a = [1, 7, 5, 6, 4, 14, 11]
for i in range(0, len(a) - 1):
    if (a[i] > a[i + 1]):
        a[i + 1] = a[i + 1] * 2
```

index	*0*	*1*	*2*	*3*	*4*	*5*	*6*
value							

- **Traversal**: An examination of each element of a list.

List mystery problem

- What element values are stored in the following list?

```
a = [1, 7, 5, 6, 4, 14, 11]
for i in range(0, len(a) - 1):
    if (a[i] > a[i + 1]):
        a[i + 1] = a[i + 1] * 2
```

index	*0*	*1*	*2*	*3*	*4*	*5*	*6*
value	1	7	10	12	8	14	22

- **Traversal**: An examination of each element of a list.

List functions

Function	Description
`append(x)`	Add an item to the end of the list. Equivalent to `a[len(a):] = [x]`.
`extend(L)`	Extend the list by appending all the items in the given list. Equivalent to `a[len(a):] = L`
`insert(i, x)`	Inserts an item at a given position. i is the index of the element before which to insert, so `a.insert(0, x)` inserts at the front of the list.
`remove(x)`	Removes the first item from the list whose value is *x*. Errs if there is no such item.
`pop(i)`	Removes the item at the given position in the list, and returns it. `a.pop()` removes and returns the last item in the list.
`clear()`	Remove all items from the list.
`index(x)`	Returns the index in the list of the first item whose value is *x*. Errs if there is no such item.
`count(x)`	Returns the number of times *x* appears in the list.
`sort()`	Sort the items of the list
`reverse()`	Reverses the elements of the list
`copy()`	Return a copy of the list.

Lists that change size

- Sometimes we don't know how big we want our list to be when our program starts
- It can be useful to create an empty list and fill it up

```
data = []
data.append("hello")
data.append("world")
print(data)                    # ['hello', 'world’]
```

- How would we insert another word in the middle?

Problem 1

- Write a function called `remove_duplicates` that takes a **sorted** list of numbers and removes any duplicates. For example, if it is called on the following list:

```
data = [-2, 1, 1, 3, 3, 3, 4, 5, 6, 78, 78, 79]
```

after the call the list should be

```
data = [-2, 1, 3, 4, 5, 6, 78, 79]
```

Looping and removing

- When you loop through a list and remove elements you change the length of the list
- This means you need to change your upper bound as you are looping

- **You must use a while loop when removing items from a list**
  - A `for i in range` loop won't work as it can't adjust when the length of the list changes

Solution Problem 1

```
def remove_duplicates(data):
    i = 0
    while i < len(data) - 1:
        if data[i] == data[i + 1]:
            data.pop(i)
        else:
            i += 1
```

List reversal

- Write code that reverses the elements of a list
  - For example, if the array initially stores: `[11, 42, -5, 27, 0, 89]`
  - Then after your reversal code, it should store: `[89, 0, 27, -5, 42, 11]`
- The code should work for a list of any size
- Hint: think about swapping various elements...

List reversal: algorithm idea

- Swap pairs of elements from the edges; work inwards:

0	1	2	3	4	5
89	0	27	-5	42	11

Swapping values

```
def main():
    a = 7
    b = 35

    # swap a with b
    a = b
    b = a

    print(a, b)
```

- What is wrong with this code? What is its output?

Swapping values

```
def main():
    a = 7
    b = 35

    # swap a with b
    a = b
    b = a

    print(a, b)
```

```
temp = a
a = b
b = temp
```

Flawed algorithm

- What's wrong with this code?

```
numbers = [11, 42, -5, 27, 0, 89]

# reverse the list
for i in range(0, len(numbers)):
    temp = numbers[i]
    numbers[i] = numbers[len(numbers) - 1 - i]
    numbers[len(numbers) - 1 - i] = temp
```

Flawed algorithm

- What's wrong with this code?

```
numbers = [11, 42, -5, 27, 0, 89]

# reverse the list
for i in range(0, len(numbers)):
    temp = numbers[i]
    numbers[i] = numbers[len(numbers) - 1 - i]
    numbers[len(numbers) - 1 - i] = temp
```

- The loop goes too far and un-reverses the array! Fixed version:

```
for i in range(0, len(numbers) // 2):
    temp = numbers[i]
    numbers[i] = numbers[len(numbers) - 1 - i]
    numbers[len(numbers) - 1 - i] = temp
```