COMP 141

Lists III

Rhodes College

Announcements

Reminder

- Program 7 - due Sunday

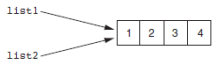
Copying Lists

- **To make a copy of a list you must copy each element of the list**
  - Two methods to do this:
    - Creating a new empty list and using a `for` loop to add a copy of each element from the original list to the new list
    - Creating a new empty list and concatenating the old list to the new empty list

Copying Lists (cont'd.)

```
list1 = [1,2,3,4]
list2 = list1
```

Figure 8-4 `list1` and `list2` reference the same list




```
list1 = [1,2,3,4]
list2 = list1
print("List 1:", list1)
print("List 2:", list2)
list1[0] = 99
print("List 1:", list1)
print("List 2:", list2)
```

Output

```
List 1: [1, 2, 3, 4]
List 2: [1, 2, 3, 4]
List 1: [99, 2, 3, 4]
List 2: [99, 2, 3, 4]
```

Code to Copy List

```
list1 = [1,2,3,4]
list2 = []
for item in list1:
    list2.append(item)
print("List 1:", list1)
print("List 2:", list2)
list1[0] = 99
print("List 1:", list1)
print("List 2:", list2)
```

Output

```
List 1: [1, 2, 3, 4]
List 2: [1, 2, 3, 4]
List 1: [99, 2, 3, 4]
List 2: [1, 2, 3, 4]
```

5

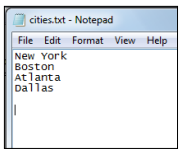
Saving Lists to a File

- **To save the contents of a list to a file:**
  - Use a `for` loop to write each element and '`\n`'
  - DON'T just typecast list to a string and write the string.

6

Writing a List to a File

```
# This program saves a list of strings to a file.

def main():
    # Create a list of strings.
    cities = ['New York', 'Boston', 'Atlanta', 'Dallas']

    # Open a file for writing.
    outfile = open('cities.txt', 'w')

    # Write the list to the file.
    for item in cities:
        outfile.write(item + '\n')

    # Close the file.
    outfile.close()

# Call the main function.
main()
```



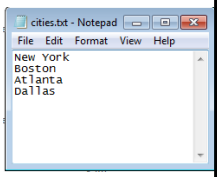
7

Reading Files into Lists

- Don't read a file into a list more than once per program.
  - This is the true power of lists – you only need to create them once and then you can use them over and over.

8

```
# This program reads a file's contents into a list.

def main():
    # Open a file for reading.
    infile = open('cities.txt', 'r')

    cities = []

    # Read the contents of the file into a list.
    for line in infile:
        line = line.rstrip()
        cities.append(line)

    # Close the file.
    infile.close()

    # Print the contents of the list.
    print(cities)

# Call the main function.
main()
```



Output

```
['New York', 'Boston', 'Atlanta', 'Dallas']
```

Comparing Lists

```
>>> list1 = ['green', 'red', 'blue']
>>> list2 = ['red', 'blue', 'green']
>>> list1 == list2
False
>>> list1 != list2
True
>>> list1 < list2
True
>>> list1 <= list2
True
>>> list1 > list2
False
>>> list1 >= list2
False
```

- To compare two lists, they must contain the same type of elements.
- The comparison uses alphabetical ordering.
  - Compares first element of each list and if they differ, this determines the outcome of the comparison.

10

Comparing Consecutive Items in List

```
def main2():
    numbers = [9, 1, 0, 2, 8, 6, 7, 5, 3, 4]
    deltas = []

    for i in range(1, len(numbers)):
        diff = numbers[i] - numbers[i-1]
        deltas.append(diff)

    print(deltas)
```

Output

```
[-8, -1, 2, 6, -2, 1, -2, -2, 1]
```

11

Practice

1. Write a program that reads in **randomNums.txt** (or a similar file with 1 number/line) and stores the values 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.

2. Write a function that prints out sums of adjacent pairs of numbers in the list
   Hint: You don't need the sliding window technique; instead, use math with list indices.

3. 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]

If you're done early, try to do part 1 without any using the built-in functions (max, min, sum)

12