Lists and Strings

Dr. Baldassano

chrisb@princeton.edu

Yu's Elite Education

Last week recap

- Loops
  - while loops
  - for loops
  - Nested loops

Putting related variables together

- Sometimes we want to keep track of a list of variables:
  - card1, card2, card3 in blackjack
- We want to put all this information into a single variable, so that we can handle any number of cards
- We'll use loops to look at all the different pieces of data

Python lists

```
scores = [9, 8.5, 4, 10]
print(scores)
```

Python lists

```
print(scores[0])
print(scores[2])
print(scores[1:3])
print(scores[-1])
print(scores[1:4:2])
print(scores[1:])
print(scores[:2])
print(scores[::2])
```

Using lists

- Creating a list: varname = [element, …]
- Accessing a list:
  - varname[i] = element i (starting from 0)
  - negative i counts from the end
  - varname[i:j] = elements i up to j (not including element j)
- Can also create list of repeated elements using * operator: `list = [True] * 10`

Lists can mix data types

```
hands = [18, 'bust', 20.0, 'bust']
print(hands)
```

Looping over lists

- We almost always want to do some action on each element in the list - use loops!

```
scores = [9, 8.5, 4, 10]
for index in range(4):
    print(index)
    print(scores[index])
    print(' ')
```

List length function

```
scores = [9, 8.5, 4, 10]
print(len(scores))
for index in range(len(scores)):
    print(scores[index])
```

List member functions

- Lists have “member” functions that are part of the list type in python
- These functions perform some action on the list
- Call like this:

```
scores = [9, 8.5, 4, 10]
scores.sort()
```

List member functions

```
scores = [9, 8.5, 4, 10]
scores.reverse()
print(scores)

scores.insert(2,7)
print(scores)

print(scores.index(7))
```

Building a list

- The append function adds a value to the end of the list

```
L = []
for n in range(2,11,2):
    L.append(n)
```

Examples

- Input numbers then sort
- Blackjack with any number of cards
- Plane route lookup
- Find minimum number in list

Removing list elements

- Two ways to remove elements:
  - Remove by index: del list[index]
  - Remove by element: list.remove(element)

```
scores = [9, 8.5, 4, 10]
del scores[0]
scores.remove(8.5)
```

Other ways to loop

```
scores = [9, 8.5, 4, 10]
```

- Looping over items:

```
for item in scores:
    print(item)
```

- Looping over both index and items:

```
for index,item in enumerate(scores):
    print(index,':',item)
```

Concatenating lists

- List1 + List2 concatenates two lists (appends one to the end of the other)

```
print([1, 2, 3] + [4, 5])

print([1, 2, 3] + 4)  # Will this work?
```

Basic math functions: sum, min, max

```
scores = [9, 8.5, 4, 10]
print('Sum:',sum(scores))
print('Min:',min(scores))
print('Max:',max(scores))
print('Avg:',sum(scores)/len(scores))
```

Examples

- Minimum 3-day hotel stay
- Sieve of Eratosthenes
- Sort by removing minimum

Read-only lists

- Lists are *mutable* – we can add/remove items
- Python's *immutable* version of lists are called "tuples"

```
days = ('Monday', 'Tuesday')
print(days[1]) # OK
days[1] = 'Wednesday' # Nope
```

When to use lists vs. tuples

- Tuples are good for hard-coding certain information into your program
  - Days of the week, Menu options…
- Otherwise almost always want to use lists

Strings: like tuples of characters

```
date = 'October 6th'
print(date[:7])
print(date[-3:])
print(date[:3] + ' ' + date[-3:])
```

String member functions

- Boolean functions:

```
date.isalpha()
date.isupper()
```

- Functions that return modified copy:

```
print(date.upper())
print(date.lower())
```

Splitting string into list

- Split member function cuts up string into a list (uses space by default)

```
text = 'Make like a banana, and split'
print(text.split())
print(text.split(','))
```

Examples

- Substitution cipher
- Class initials

Assignment

- Problem #8 from Project Euler.net
- https://projecteuler.net/index.php?section=problems&id=8
- Find the thirteen adjacent digits in the 1000-digit number that have the greatest product. What is the value of this product?
- Python file with 1000-digit number on my website