

PIC 16, Winter 2018

Lecture 2W: Data Structures
Wednesday, January 17, 2018
Matt Haberland

Announcements

- Check out videos posted in 2W on CCLE

Questions

- How are the different sequence types implemented, and what are the implications?
 - List
 - Set
 - Dict
- What is the difference between $x += y$ and $x = x + y$?
 - For immutable types
 - For mutable types

Intended Learning Outcomes

- By the end of today, I want you all to be able to:
 - perform lexicographical comparison, and
 - read and write list, set, and dictionary comprehensions.

Lexicographical Comparison

- Compare:

- $1 < 2$
- $'a' < 'b'$
- $'ba' < 'ab'$
- $'ab' < 'ba'$
- $'aa' < 'ab'$
- $'a' < 'ab'$
- $('aa', 'aa') < ('aa', 'aaa')$
- $('a', (1, 2), 'd') < ('a', (1, 2, 3))$

List Comprehension

- Write a list comprehension that generates a list of letters from 'a' to 'z' (without typing them all).
 - `ord` function accepts a single-character string and returns the corresponding ASCII character code
 - `chr` function accepts an ASCII character code and returns a single-character string
 - First try writing a for loop, then convert to list comprehension

List Comprehension

- Useful for both:

```
alphabet_numbers = range(ord('a'), ord('z')+1)
```

- for loop:

```
alphabet = []  
for i in alphabet_numbers:  
    alphabet.append(chr(i))
```

Initialization
How to loop
What to append

- List comprehension

```
alphabet = [chr(i) for i in alphabet_numbers]
```

How to loop
What to append

Dictionary Comprehension




- Write a dictionary comprehension that maps numbers 1 – 26 to the corresponding letter of the alphabet
 - i.e. {1: 'a', 2: 'b', ...}

Dictionary Comprehension



- Useful for both:

```
alphabet_numbers = range(ord('a'), ord('z')+1)
```

- for loop:

```
alphabet = {}  Initialization  
for i in alphabet_numbers:  How to loop  
    alphabet[i - ord('a') + 1] = chr(i)  pair to add
```

- Dictionary comprehension

```
alphabet = {  pair to add  How to loop  
    i-ord('a')+1: chr(i) for i in alphabet_numbers }
```

Questions?

Quiz 2W

- Please complete Quiz 2W on CCLE
- Please *do not* refer to the preparation document or other materials. Answers are to come from your brain only.

Activity

- Work on Assignment 1F,
- Start Assignment 2W, or

Consider the [Sieve of Eratosthenes](https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes).

https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes

Consider each of the four data structures (list, tuple, set, dict) separately.

Can the sieve be implemented using that data structure?

If so, implement the sieve in Python with that data structure.

Hint: Yes, it can be implemented using a list. We can “mark” elements of the list by setting them to `False`.

A swap function

```
a = 1
```

```
b = 2
```

```
def swap(a, b):  
    a, b = b, a
```

```
swap(a, b)
```

```
print "a:", a  
print "b:", b
```

```
a: 1  
b: 2
```

What prints?

A swap function

```
a = 1
```

```
b = 2
```

```
def swap(a, b):  
    return b, a
```

```
a, b = swap(a, b)
```

```
print "a:", a  
print "b:", b
```

```
a: 2  
b: 1
```

What prints?

References passed by value

- In C++, you often:
 - passed a variable by value, creating a copy
 - passed a variable by reference, creating an alias for the variable inside the function
- In Python, all variables are just references to objects
- In Python, we always pass these references “by value”
 - We create a copy of the reference inside the function
 - We don’t create a copy of the object the reference refers to
- This is like passing a pointer by value in C++