Introduction to Python Lecture

Session 1, Sept. 27

Review

- math (`+`, `-`, `/`, `*`)
- `type()`
- variables
- strings
- booleans
- `if/elif/else`
- functions

Math Review

```
>>> 5 + 6
11

>>> -6
-6

>>> ((2+5)*3)/7
3
```

type()

```
>>> type(5)
<type 'int'>

>>> type(True)
<type 'bool'>

>>> type("Hello!")
<type 'str'>
```

Variables

```
>>> variable = 5
>>> variable
5

>>> variable = "varying"
>>> variable
'varying'
```

Strings

```
>>> "this is a string" + " and another"
'this is a string and another'

>>> "string" * 3
'stringstringstring'
```

Booleans

```
>>> 5 > 3
True

>>> 'a' in 'apple'
True

>>> 'Windows' == 'Best Operating System'
False
```

if/elif/else

```
>>> today = 'Saturday'
>>> if today == 'Saturday':
...     print('Pizza lunch!')
... else:
...     print("No pizza :'(")
...
Pizza lunch!
```

Functions

```
>>> def omguw(year):
...   print("Your favorite on-campus gossip
            source since " + year)
...

>>> omguw("2008")
Your favorite on-campus gossip source since 2008
```

Lists!

- Purpose
- Initialization
- `len()` review
- Using lists
- “Slicing” lists
- Strings are like lists!

What is a list?

- We have only worked with “atomic” data types thus far
- This is somewhat limiting
- We have 10 provinces and 3 territories:
  - province_1 = "Alberta"
  - province_2 = "British Columbia"
  - …

The empty list

```
>>> nothing = []

>>> nothing
[]
```

What is a list?

```
>>> provinces = ['AB', 'BC', 'MB',
'NB', 'NL', 'NS', 'NT', 'NU', 'ON',
'PE', 'QC', 'SK', 'YT']

>>> provinces
['AB', 'BC', 'MB', 'NB', 'NL', 'NS',
'NT', 'NU', 'ON', 'PE', 'QC', 'SK',
'YT']
```

`len()` review for lists

```
>>> provinces = ['AB', 'BC', 'MB',
'NB', 'NL', 'NS', 'NT', 'NU', 'ON',
'PE', 'QC', 'SK', 'YT']

>>> len(provinces)
13
```

Using lists

```
>>> provinces = ['AB', 'BC', 'MB', 'NB',
'NL', 'NS', 'NT', 'NU', 'ON', 'PE', 'QC',
'SK', 'YT']

>>> # Accessing an item in a list
>>> provinces[3]
'NB'
>>> provinces[0]
'AB'
>>> provinces[13]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: list index out of range
```

Using lists

```
>>> provinces = ['AB', 'BC', 'MB', 'NB',
'NL', 'NS', 'NT', 'NU', 'ON', 'PE', 'QC',
'SK', 'YT']

>>> # Adding an element to a list
>>> # (Take over the state of Washington)
>>> provinces.append('WA')
>>> provinces[13]
'WA'
>>> provinces[-1]
'WA'
```

Using lists

```
>>> provinces = ['AB', 'BC', 'MB', 'NB',
'NL', 'NS', 'NT', 'NU', 'ON', 'PE', 'QC',
'SK', 'YT', 'WA']

>>> # Changing an element in a list
>>> # (Decide we want Alaska instead)
>>> provinces[13]
'WA'
>>> provinces[13] = 'AK'
>>> provinces[13]
'AK'
```

“Slicing” lists

```
>>> provinces = ['AB', 'BC', 'MB', 'NB',
'NL', 'NS', 'NT', 'NU', 'ON', 'PE', 'QC',
'SK', 'YT', 'AK']

>>> # Return a portion of a list
>>> # Syntax: list[start:end]
>>> provinces[3:7]
['NB', 'NL', 'NS', 'NT']
>>> provinces[11:]
['SK', 'YT', 'AK']
>>> provinces[:-1]
['AB', 'BC', 'MB', 'NB', 'NL', 'NS', 'NT',
'NU', 'ON', 'PE', 'QC', 'SK', 'YT']
```

Strings are (kind of) like lists

```
>>> listy_string = 'hello'

>>> listy_string[3]
'l'
>>> listy_string[:-1]
'hell'

>>> # Beware!
>>> listy_string.append(' world')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'str' object has no
attribute 'append'
```

Flow control

- `for` loops
- `while` loops
- Infinite loops
- `break`
- `raw_input()`

Basic `for` loop

```
>>> for province in provinces:
...   print("I don't live in " + province)
...
I don't live in AB
I don't live in BC
[...]
I don't live in AK
```

`for` loop with `if`

```
>>> for province in provinces:
...   if province == 'ON':
...     print("I live in " + province + "!")
...   else:
...     print("I don't live in " + province)
...
I don't live in AB
[...]
I don't live in NU
I live in ON!
I don't live in PE
[...]
I don't live in AK
```

Nested `for` loops

```
>>> provinces = ['AB', 'BC', 'MB', 'NB',
'NL', 'NS', 'NT', 'NU', 'ON', 'PE', 'QC',
'SK', 'YT', 'AK']

>>> for province in provinces:
...     for letter in province:
...         if letter == 'A':
...             print(letter)
...
A
A
```

range()

```
>>> # range() builds a list of ints

>>> range(10,15)
[10, 11, 12, 13, 14]
>>> for num in range(0, 5):
...     print(num)
...
0
1
2
3
4
```

Basic `while` loop

```
>>> i = 5
>>> while (i < 8):
...     i += 1
...     print(i)
...
6
7
8
```

Infinite loops

```
>>> while (true):
...

>>> # Why would we ever do this?!
```

raw_input() and break

```
>>> secret_word = 'python'
>>> guess = raw_input()
ghost
>>> guess
'ghost'
>>> while (True):
...   print('Guess my secret word: ')
...   guess = raw_input()
...   if guess == special_word:
...     print('You got it!')
...     break
...
Guess my secret word:
foo
Guess my secret word:
python
You got it!
```

Break time!

Practice Problem:

Given the following list of grades, determine the class average.

(Hint: *Use loops!*)

```
grades = [86, 71, 91, 62, 85, 86, 72,
58, 65, 77, 54, 53, 56, 91, 93, 58,
80, 74, 75, 64]
```

Dictionaries!

- Purpose
- Initialization
- Using lists
- `keys()` and `values()`

What is a dictionary?

- Lists only store a sequence of data
- Often, we want to associate “keys” with “values”
- How would we actually store a class' test scores?
  - johnny = 85
  - yifan = 79
  - gloria = 90
  - ...

The empty dictionary

```
>>> nothing = {}

>>> nothing
{}
```

What is a dictionary?

```
>>> # Also called “association
>>> # lists”, “hashes”, “maps”, etc

>>> grades = {
   'johnny': 85,
   'yifan':  79,
   'gloria': 90,  # trailing comma
}
>>> grades
{'yifan': 79, 'gloria': 90, 'johnny': 85}

>>> grades['yifan']
79
```

Using dictionaries

```
>>> grades = {
   'johnny': 85,
   'yifan':  79,
   'gloria': 90,
}

>>> # Add an item to a dictionary
>>> grades['salem'] = 53
>>> grades
{'yifan': 79, 'salem': 53, 'gloria': 90,
'johnny': 85}
```

Using dictionaries

```
>>> # Be careful with KeyValue errors

>>> grades['elana']
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'elana'

>>> # Safer:
>>> grades.get('elana')
>>>
```

Using dictionaries

```
>>> grades = {
   'johnny': 85,
   'yifan':  79,
   'gloria': 90,
   'salem':  53,
}

>>> # Change a value in the dictionary
>>> grades['salem']
53
>>> grades['salem'] = 100
>>> grades['salem']
100
```

keys() and values()

```
>>> grades = {
   'johnny': 85,
   'yifan':  79,
   'gloria': 90,
   'salem':  100,
}

>>> # Get a list of the keys in the dict
>>> grades.keys()
['yifan', 'salem', 'gloria', 'johnny']

>>> # Get a list of the values in the dict
>>> grades.values()
[79, 100, 90, 85]
```

keys() and values()

```
>>> # Class average
>>> grade_list = grades.values()
>>> sum = 0

>>> for grade in grade_list:
...     sum += grade

>>> sum
354
>>> print(sum / len(grade_list))
88
```

Modules

- Purpose
- Built-ins
- Importing
- The `random` library

What is a module?

- A unified body of code someone else wrote
- We don't want to have to write everything ourselves!
  - Philosophy behind free software
- **Libraries:** contain many modules
- Python has many built-in modules

Built-in Modules and More

- Built-in
  - `random`: random numbers
  - `urllib`: utilities for working with URLs

```
>>> import urllib
>>> urllib.quote('url.with.a.space/here yay')
'url.with.a.space/here%20yay'
```

- Other
  - `matplotlib`: draw graphs and plots!
  - List: https://wiki.python.org/moin/UsefulModules

Using a module

```
>>> random.randint(0,5)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'random' is not defined

>>> # Make sure we have the code available!
>>> import random
>>> random.randint(0,5)
5
>>> random.randint(0,5)
4
```

Using a module

```
>>> provinces = ['AB', 'BC', 'MB', 'NB', 'NL', 'NS',
'NT', 'NU', 'ON', 'PE', 'QC', 'SK', 'YT', 'AK']

>>> # Documentation for modules online!
>>> # https://docs.python.org/2/library/random.html
>>> import random
>>> random.choice(provinces)
'NL'
>>> random.randint(0,5)
'ON'
```

Putting it all together

- Demo of `provincial_capitals.py`
  - Follow along on your own computer
  - Make sure to ask questions!

- Break for lunch afterwards!
- Return from lunch for 12:45 PM