Ice sculptures displayed at the annual Ice and Snow festival in Harbin. Fairy tale palaces, towering pagodas, and even an Egyptian Sphynx -- all carved from ice -- are among the sights at the Harbin International Ice and Snow Sculpture Festival. (Stringer/AFP/Getty Images)

Functions

Lecture 13 - COMP 364
February 12, 2010, updated 2012
Mathieu Perreault

Announcements

- First part of midterm exam, Lecture[0-6].pdf
    - Wednesday Feb. 15th in class.
- Second part of midterm, Lecture([7-9]|1[0-4]).pdf (7 to 14).
    - Friday Feb 17th in the lab (TR3070).

Objects have functions defined on them

- In Python, **almost everything** is an object
    - ??

Objects have functions defined on them

- In Python, **almost everything** is an object
  - List, String, File, Set, Tuple, Dictionary, custom-built objects, etc.
- Python objects have functions defined on them that follow the pattern:
  <object>.<function>([arguments])

```
mylist = []
mylist.append(3)
mylist.insert(0, “Hello”)
```

String Functions

- Always check the documentation to see which functions are available
    - http://docs.python.org/library/string.html#deprecated-string-functions
- Advanced: Python also has introspection features: dir(<object>) and __doc__ can help you sometimes.

String Functions

- x.strip() - remove whitespace on either end of a string
    - x.lstrip(), x.rstrip()
- x.replace('foo','bar')
    - "foobar".replace('foo','bar') -> 'barbar'

Checking for content

- *x.startswith(y)* - does string x start with string y?
- *x.endswith(y)* - does string x end with string y?
- *y in x* - is the string y found anywhere in x?
- x.find(y) - finds the earliest instance of y in x. Returns the position.

Making a string into a list!

- x.split() - return a list of all substrings in x separated by spaces
  - “x y z”.split() -> [“x”,”y”,”z”]
- x.split(‘,’)
  - “9606,core,Homo Sapiens”.split(‘,’) -> [“9606”, “core”, “Homo Sapiens”]

Regular expressions

- *import re* - re is the regular expression module (http://docs.python.org/library/re.html)
    - *re.findall(‘<regexp>’,x)* - find all places where the regular expression matches something in x
        - Returns a list!
- *re.search(‘<regexp>’,x)* - find exactly one place where the regular expression matches something in x, where x is a string.
    - Returns a *MatchObject*

The MatchObject

- m.groups(0) - return the matched string
- m.groups(i) - return the ith group in the match

- Examples:
  - *m = re.search('([0-9]+)([a-z]+)\.py','test33index.py')*
  - *m.groups(0) -> '33index.py'*
  - *m.groups(1) -> '33'*
  - *m.groups(2) -> 'index'*

Extra: string formatting

- When you want to print complex strings, use string substitution
- Main substitution types: %s (string), %d (digits/integers), %f (float)

```
print “This sequence is %d nucleotides long” % len(line)
print “For taxon ID %d, Species name is: %s” % (mylist[0],
 mylist[1])
```