Week 3: Python

python

What is Python?

- not just a scripting language
- object-oriented language
    - classes
    - member functions
- compiled and interpreted
    - Python code is compiled to bytecode
    - bytecode is interpreted by Python interpreter
- not as fast as C but easy to learn, read and use
- very popular at Google and others!

Optparse Library (deprecated; replaced by argparse)

- powerful library for parsing command-line options
- `$ python randline -n5 filetext`
    - handles everything after “randline"

Argument

- string entered on the command line and passed in to the script
- elements of `sys.argv[1:]` (`sys.argv[0]` is the name of the program being executed)
- Ex: `filetext`

Option

- an argument that supplies extra information to customize the execution of a program
- Ex: `-n`

Option Argument

- an arguent that follows an option and is closely associated with it
- consumed from argument list when the option is
- Ex: `-n5`

Python Lists

- common data structure in Python
- python list is like a C array but much more

    - **dynamic (mutable)**: expands as new items are added
    - **heterogeneous**: can hold objects of different types
- accessing elements
    - `list_name[index]`

```
## List Example ##
>>> t = [123, 3.0, ‘hello!’]
>>> print t[0]
123
>>> print t[1]
3.0
>>> print t[2]
hello!
```

```
## Example: Merging Lists ##
>>> list1 = [1, 2, 3, 4]
>>> list2 = [5, 6, 7, 8]
>>> merged_list = list1 + list2
>>> print merged_list
## Output: [1,2,3,4,5,6,7,8]
```

Python Dictionaries

- essentially a hash table
    - provides key-value (pair) storage capability
- instantiation:
    - dict = {}
    - this creates an **empty** dictionary
- keys are unique, values are not!
    - keys must be immutable (strings, numbers, tuples)

```
## Dictionary Example ##
dict = {}
dict[‘hello’] = “world”
print dict[‘hello’]
World

dict[‘power’] = 9001
if (dict[‘power’] > 9000):
    print “Its over “, dict[‘power’]
    ## Its over 9001

## Deleting dictionary (elements)
del dict[‘hello’]
del dict
```

For Loops

```
list = [‘Mary’, ‘had’, ‘a’, ‘little’, ‘lamb’]
```

```

## Example 1 ##
for i in list:
    print i

# Output:
# Mary had a little lamb (on separate lines)

## Example 2 ##
for i in range(len(list)):
    print i

# Output: 0 1 2 3 4 (on separate lines)

## To print not on separate lines,
## sys.stdout.write()
```

Indentation

- Python has no braces or keywords for code blocks
    - C delimiter: {}
    - bash delimiter:
        - `then… else… fi` (if statements)
        - `do… done` (while, for loops)
- indentation makes all the difference
    - tabs change code’s meaning!