Lecture 14
Python: strings, file handling

Course: Practical Bioinformatics (BIOL 4220)
Instructor: Michael Landis
Email: michael.landis@wustl.edu

Lecture 14 outline

Last time: containers

This time: Python (2 of 4)

Python

- strings
- filesystem
- input/output

Strings as containers

Substrings may be extracted from strings using the index operator, *[ ]*

```
>>> x = 'Cookie Monster'
>>> x[0] # return first character
'C'
>>> x[0:6] # return characters 0 to 6
'Cookie'
>>> x[:6] # return up to character before index 6
'Cookie'
>>> x[7:] # return character at index 7 through end
'Monster'
>>> x[0:2] + x[7:9] # concatenate two substrings
'CoMo'
>>> x[ [0,1,2] ] # cannot index string with an index list
Traceback (most recent call last):
   File "<stdin>", line 1, in <module>
TypeError: string indices must be integers
```

Strings methods

A ***method*** is a function that is associated with a variable, and operates on that variable

All Python strings are equipped with a suite of powerful built-in string-manipulation methods

```
>>> x = 'my shift key is broken'
>>> x.upper()
'MY SHIFT KEY IS BROKEN'
>>> help(x.upper)

Help on built-in function upper:

upper(...) method of builtins.str instance

        S.upper() -> str

        Return a copy of S converted to uppercase.
(END)
```

Strings methods

Change letters between upper and lowercase

```
>>> x = 'hey, do penguins have DNA?'
>>> x.upper() # all letters to uppercase
'HEY, DO PENGUINS HAVE DNA?'
>>> x.lower() # all letters to lowercase
'hey, do penguins have dna?'
>>> x.title() # 1st letter of each word to uppercase
'Hey, Do Penguins Have Dna?'
>>> x.capitalize() # capitalize 1st letter, rest lowercase
'Hey, do penguins have dna?'
>>> x.swapcase() # change upper to lowercase, and vice versa
'HEY, DO PENGUINS HAVE dna?'
```

Strings methods

Reformat characters flanking a string

```
>>> x = ' a long pause '
>>> x
' a long pause '
>>> x.strip() # remove all flanking whitespace
'a long pause'
>>> x.lstrip() # remove all whitespace on left
'a long pause '
>>> x.rstrip() # remove all whitespace on right
' a long pause'
>>> x.center(30,'.') # create length-30 string, buffered with .
'....... a long pause .......'
>>> f = '7'
>>> f.zfill(3) # create length-3 string, buffered on left with 0
'007'
```

Strings methods

Test string properties, return boolean values

```
>>> x = 'Hello' # create string
>>> x.isalpha() # does x only contain alphabetical chars?
True
>>> 'Hello'.isalpha() # call isalpha() method against string value
True
>>> 'hello'.islower() # are all letters in string lowercase?
True
>>> 'HELLO'.isupper() # are all letters in string uppercase?
True
>>> 'h3ll0'.isalnum() # are all characters alphanumeric?
True
>>> '63110'.isdigit() # are all characters numbers?
True
>>> ' \t\n'.isspace() # are all characters whitespace?
True
>>> 'Hello'.startswith('He') # does string start with supplied string?
True
>>> 'Hello'.endswith('lo') # does string end with supplied string?
True
```

Strings methods

Search for patterns within the string

```
>>> x = 'Shelly sells seashells'
>>> x.count('ell') # how many times does substring appear?
3
>>> x.find('ell') # find index of first occurrence of substring
2
>>> x[2:]
'elly sells seashells'
>>> x.rfind('ell') # find index of last occurrence of substring
18
>>> x[18:]
'ells'
>>> x.find('seashore') # find returns -1 if substring not found
-1
```

Strings methods

Join and split strings

```
>>> x = 'together forever'
>>> x.split(' ') # tokenize string into list using delimter
['together', 'forever']
>>> x.replace('er','a') # replace all instances of substring
'togetha foreva'
>>> y = ['b','n','n','j','m']
>>> 'a'.join(y) # use first string as "glue" to concatenate list
'bananajam'
>>> z = 'upstairs\ndownstairs'
>>> z.splitlines(keepends=True) # split string using '\n' delimiter
['upstairs\n', 'downstairs']
```

Chaining methods

Many string methods will return string values upon completion;
the returned value can itself call another method!
This is sometimes called ***method chaining***.

```
>>> x = 'I am not a crook'
>>> x.upper()
'I AM NOT A CROOK'
>>> x.isupper()
False
>>> x.upper().isupper()
True
>>> 'I am not a crook'[11:].upper().lower().islower()
True
```

Formatting strings

Substitute variables into strings with {*x*} notation

```
>>> mood = 'love'
>>> food = 'donuts'
>>> print('I ' + mood + ' to eat ' + food + '!')
I love to eat donuts!
>>> print(f'I {mood} to eat {food}')
I love to eat donuts!
>>> print('I {x} to eat {y}'.format(x='love', y='donuts')
I love to eat donuts!
```

Hundreds of ways to format numerical variables

```
>>> '{:06.2f}'.format(3.1415926535897793) '003.14'
>>> import datetime
>>> x = datetime.datetime(2020, 11, 7, 12, 39)
>>> '{:%Y-%m-%d %H:%M}'.format(x)
'2020-11-07 12:39'
```

Listing filesystem objects

List all files and directories

```
>>> import os
>>> path = '/home/data_analysis/netflix'
>>> os.listdir(path)
['file.txt', 'docs', 'data']
```

List all files and directories; supports wildcard filters

```
>>> import glob
>>> path = '/home/data_analysis/netflix'
>>> glob.glob(path + "/*.txt")
['file.txt']
```

Function “walks” through part of filsystem and saves files vs. directories

```
>>> import os
>>> path = '/home/data_analysis/netflix'
>>> for root, dirs, files in os.walk(path):
... for name in files:
... print(os.path.join(root, name))
... for name in dirs:
... print(os.path.join(root, name))
```

Reading from file

Call *open(filename, 'r')* to begin reading a file;
use a for-loop to iterate over each line in the file

```
>>> dirname = '/home/mlandis/'
>>> filename = dirname + 'test.txt'
>>> s = ''
>>> # open the file for reading ('r')
>>> f = open(filename, 'r')
>>> for line in f:
... s += line + '\n'
...
>>> f.close()
>>> print(s)
upstairs
downstairs
```

Writing to file

Call *open(filename, 'w')* to begin writing to a file; append new content to the file with *f.write(text)*

```
>>> dirname = '/home/mlandis/'
>>> filename = dirname + 'test.txt'
>>> s = ''
>>> # open the file for writing ('w')
>>> f = open(filename, 'w')
>>> N = 3
>>> for i in range(N):
... f.write(f'{i+1} of {N}\n')
...
>>> f.close()
>>> quit()
```

code

```
$ cat /home/mlandis/test.txt
1 of 3
2 of 3
3 of 3
```

shell

Example script

```
# filesystem
lab_dirname = '/home/mlandis/labs/lab_09a/'
in_filename = lab_dirname + 'input.txt'
out_filename = lab_dirname + 'output.txt'
# read in file
in_file = open(in_filename, 'r') x = {}
for i,line in enumerate(in_file):
    # get all fields per row
    fields = line.split(',')
    # ignore header
    if i > 0:
        x[i] = []
    # collect fields
    for f in fields:
        x[i].append(f)

in_file.close()

# write out file
out_file = open(out_filename, 'w')
for i in range(len(x)):
    # row elements -> tab-separated string
    row = '\t'.join(x[i])
    # write each row to file
    out_file.write(row + '\n')

out_file.close()
```

Overview for Lab 14