Programming for Biologists

Streams

- A stream is a temporally ordered sequence of indefinite length
  - Keyboard entries are a stream of characters that Python is processing
  - Data residing in files is also accessed as a stream
    - We think of files as data on some media, but in programming it can be abstract to include anything from an external source
- Python allows you to read files in many different modes
- To create a file object, simply use the open command:

```
f = open(path, mode)
```

Different modes

Value	Mode	Interpretation
t	Text (default)	Characters or Strings
b	Binary	bytes

Value	Initial Position	Read	Write
r	Beginning	Y	N
w	Beginning	N	Y
a	End	N	Y
r+	Beginning	Y	Y
w+	Beginning	Y	Y
a+	End	Y	Y

File operations

Operation	Interpretation
output = open(r'C:\spam', 'w')	Create output file ('w' means write)
input = open('data', 'r')	Create input file ('r' means read)
input = open('data')	Same as prior line ('r' is the default)
aString = input.read()	Read entire file into a single string
aString = input.read(N)	Read up to next N characters (or bytes) into a string
aString = input.readline()	Read next line (including \n newline) into a string
aList = input.readlines()	Read entire file into list of line strings (with \n)
anyFile.seek(N)	Change file position to offset N for next operation
for line in open('data'):	use line File iterators read line by line
open('f.txt', encoding='latin-1')	Python 3.0 Unicode text files (str strings)
open('f.bin', 'rb')	Python 3.0 binary bytes files (bytes strings)

File operations

Operation	Interpretation
output.writelines(aList)	Write all line strings in a list into file
output.close()	Manual close (done for you when file is collected)
output.flush()	Flush output buffer to disk without closing
output.write(aString)	Write a string of characters (or bytes) into file

Learning Python, Table 9-2, p. 232

Input/Output: working with files

You can create a file object to work with by assigning the open function to a name:

```
input = open('sequences.fa','r')          # open a fasta file for reading
input = open('./sequences.fa')            # same as above
output = open('processed_data.txt','w') # open outfile for writing
```

The with statement

This statement is used to open and name a file, then automatically close the file (regardless of any errors that may occur during execution of its statements).

You can also open a list of files, e.g. to read from one file and write to another

```
with open(path,mode) as name:
    statements-using-name
```

```
with open(path1,mode1) as name1, open(path2,mode2) as name2, ... :
    statements-using-name
```

File data are interpreted as strings!

- Whenever you read or write a line in a file from Python, all file text takes the form of strings.
- Therefore, if you want to work with numbers in a data file, you have to convert string data to numbers.
- Similarly, you have to send output as formatted strings.
- String conversion can be done using e.g. int, float, str, and string formatting expressions.