

## One minute responses

---

- Are we not doing any more Python after next week?
- I think being able to fully go through one example is helpful.
- Building on previous learning in the Python programming would be great (e.g. use the same base program a few classes in a row so I don't spend time writing basic error checking code).
- It was very helpful to have feedback about the homework—it would be great to have that every time we get an assignment back
- Could you give some tips to return an elegant chart output for Python problems (and HW solutions)

## **Useful sources of Python information**

---

- Tutorial: <http://docs.python.org/tut/> (by the author of Python)
- References: <http://www.python.org/doc/>

## Making the solution to HW4 look elegant

---

```
import sys
filehandle = open(sys.argv[1], "r")

#discard the first line
garbage = filehandle.readline()

lines = filehandle.readlines()
filehandle.close()

names = []
seqs = []
for line in lines :
    names.append(line[0:10].rstrip())
    seqs.append(line[10:].rstrip())

#table columns will be 10 characters wide
colwidth = 10
```

```
#the first column has no title
header = "\t"
header = header.expandtabs(colwidth)
for name in names :
    centered = name.center(colwidth)
    header += "%10s" %(centered)
print header

import dna_fx
for index1 in range(0,len(names)) :
    #left-justify the names, center the numbers
    row = "%-10s" %(names[index1])
    for index2 in range(0,len(names)) :
        value = str(dna_fx.distance(seqs[index1],seqs[index2]))
        centered = value.center(colwidth)
        row += "%10s" %(centered)
    print row
```

## **Review class today**

---

- In theory, today was supposed to be BioPython
- The lab staff are struggling to install it—hopefully by Tuesday
- Today we'll do a set of linked Python problems instead

# Problem 1: Translating RNA to protein

---

- I have created a Python dictionary containing the genetic code
- Format:
  - Filename is `code.py`
  - Dictionary name is `rna_to_aa`
  - Key is an RNA codon, upper case, as a string: "UCC"
  - Entry is a three-letter amino acid abbreviation or "Stp" for a stop codon
- Import this dictionary into a program of your own
- Use it to translate a file containing codons; a sample file is `codon_rna.txt`
- The file contains only one line, with one or more complete codons on it
- Your program should read this file and print a protein translation

- Program should work for upper or lower case RNA

## Problem 2: Translating DNA to protein

---

- WITHOUT CHANGING THE DICTIONARY, allow your program to translate DNA to protein as well as RNA
- A sample file is codon\_dna.txt
- Make sure it continues to work on codon\_rna.txt
- Make sure it still works for upper and lower case

## **Problem 3: Reverse translation (discussion)**

---

- Could we reverse this dictionary to make a protein to RNA dictionary?
- If so, how?
- If not, why not?

## Problem 4: Sequence not broken into codons

---

- Suppose our sequence is not divided into codons with spaces between them
- Modify your program so that it can handle a continuous DNA sequence
- If there are not enough bases to finish the last codon, rather than printing an amino acid, print “???”

# Problem 1 solution

---

```
import sys
filename = sys.argv[1]
filehandle = open(filename,"r")
```

```
rnaseq = filehandle.readline()
rnaseq = rnaseq.upper()
```

```
import code
codons = rnaseq.split()
for codon in codons :
    print code.rna_to_aa[codon],
```

```
python p1.py codon_rna.txt
```

```
Met Pro Val Val Stp Stp Asn Asn Ala Glu Cys
```

## Problem 2 solution

---

Short way:

```
rnaseq = rnaseq.upper()
rnaseq = rnaseq.replace("T","U")
```

Not so short way:

```
rnaseq = rnaseq.upper()

# transform string into a list
# which can be changed in place
rnalist = list(rnaseq)
for index in range(0,len(rnalist)) :
    if rnalist[index] == "T" :
        rnalist[index] = "U"
# transform list back into string
rnaseq = "".join(rnalist)
```

## Problem 3 thoughts

---

- Keys in a dictionary have to be unique
- This means we can't have a dictionary where the amino acid is the key and the codon is the entry
- We could have a dictionary where the amino acid is the key and a list of codons is the entry
- Is that useful? Depends on why we wanted it!

## Problem 4 solution

---

```
import sys
filename = sys.argv[1]
filehandle = open(filename,"r")

rnaseq = filehandle.readline()
rnaseq = rnaseq.upper()
rnaseq = rnaseq.replace("T","U")

import code
codons = []
extrabases = False
for index in range(0,len(rnaseq),3) :
    if index+2 < len(rnaseq) :
        codons.append(rnaseq[index:index+3])
    else :
        extrabases = True
for codon in codons :
```

```
    print code.rna_to_aa[codon],  
if extrabases :  
    print "???"
```