```
1  from itertools import izip, chain, repeat, count
2  import re, operator
3  import pickle as p
4  from string import strip, maketrans, translate
5  from pylab import *
6
7
8  stopcodons = [('T','A', 'G'), ('T', 'A', 'A'), ('T', 'G', 'A')]
9
10 def grouper(n, iterable, padvalue=None):
11     return izip(*[chain(iterable, repeat(padvalue, n-1))]*n)
12
13 def sanitycheck(seq):
14     """proves that seq is a valid DNA sequence without any whitespace"""
15     foldr = reduce
16     return foldr(operator.__and__, map(lambda x: x in ['A','T', 'C', 'G'], seq
   ), True)
17
18 def reverseComplement(DNA):
19     table = maketrans('ACTG', 'TGAC')
20     return DNA.translate(table)[::-1]
21
22 def findORF(seq):
23     print sanitycheck(seq)
24     oRFStart = []
25     oRFLen = []
26     inORF = False
27     for offset in xrange(3):
28         for (ncod, codon) in izip(count(), grouper(3, seq[offset:])):
29             if not inORF:
30                 if codon == ('A', 'T', 'G'):
31                     inORF = True
32                     oRFStart.append(ncod)
33             if inORF:
34                 if codon in stopcodons:
35                     oRFLen.append(ncod-oRFStart[-1])
36                     inORF = False
37
38     rcseq = reverseComplement(seq)
39     inORF = False
40     for offset in xrange(3):
41         for (ncod, codon) in izip(count(), grouper(3, rcseq[offset:])):
42             if not inORF:
43                 if codon == ('A', 'T', 'G'):
44                     inORF = True
45                     oRFStart.append(ncod)
46             if inORF:
47                 if codon in stopcodons:
48                     oRFLen.append(ncod-oRFStart[-1])
49                     inORF = False
50
51
```

```
52      p.dump((oRFStart, oRFLen), file("ORF.txt", "w"))
53
54  def draw():
55      (oRFStart, oRFLen) = p.load(file("ORF.txt"))
56      cleanoRFLen = [x for x in oRFLen if x >= 100]
57      hist(cleanoRFLen, 75)
58      title('histogram of ORF lengths')
59      ylabel('frequency of occurrence')
60      xlabel('length of ORF in nt')
61      show()
62      print max(cleanoRFLen)
63      #print oRFStart
64      #print oRFLen
65
66  if __name__=="__main__":
67      seq = "".join(map(strip, file("ecoli.fasta").readlines()))
68      findORF(seq)
69      draw()
70
```