



✓ Congratulations! You passed!

TO PASS 70% or higher

Keep Learning

GRADE
100%

Programming Homework 4

LATEST SUBMISSION GRADE

100%

1. In a practical, we saw the `scs` function (copied below along with `overlap`) for finding the shortest common superstring of a set of strings.

1 / 1 point

```
1 def overlap(a, b, min_length=3):
2     """ Return length of longest suffix of 'a' matching
3         a prefix of 'b' that is at least 'min_length'
4         characters long. If no such overlap exists,
5         return 0. """
6     start = 0 # start all the way at the left
7     while True:
8         start = a.find(b[:min_length], start) # look for b's suffix in a
9         if start == -1: # no more occurrences to right
10            return 0
11        # found occurrence; check for full suffix/prefix match
12        if b.startswith(a[start:]):
13            return len(a)-start
14        start += 1 # move just past previous match
15
16 import itertools
17
18 def scs(ss):
19     """ Returns shortest common superstring of given
20         strings, which must be the same length """
21     shortest_sup = None
22     for ssperm in itertools.permutations(ss):
23         sup = ssperm[0] # superstring starts as first string
24         for i in range(len(ss)-1):
25             # overlap adjacent strings A and B in the permutation
26             olen = overlap(ssperm[i], ssperm[i+1], min_length=1)
27             # add non-overlapping portion of B to superstring
28             sup += ssperm[i+1][olen:]
29             if shortest_sup is None or len(sup) < len(shortest_sup):
30                 shortest_sup = sup # found shorter superstring
31     return shortest_sup # return shortest
```

It's possible for there to be multiple different shortest common superstrings for the same set of input strings. Consider the input strings ABC, BCA, CAB. One shortest common superstring is ABCAB but another is BCABC and another is CABCA.

What is the length of the shortest common superstring of the following strings?

CCT, CTT, TGC, TGG, GAT, ATT

✓ Correct

2. How many different shortest common superstrings are there for the input strings given in the previous question?

1 / 1 point

Hint 1: You can modify the `scs` function to keep track of this.

Hint 2: You can look at [these examples](#) to double-check that your modified `scs` is working as expected.

✓ Correct

3. Download this FASTQ file containing synthetic sequencing reads from a mystery virus:

1 / 1 point

https://d28rh4a8wg0iu5.cloudfront.net/ads1/data/ads1_week4_reads.fq

All the reads are the same length (100 bases) and are exact copies of substrings from the forward strand of the virus genome. You don't have to worry about sequencing errors, ploidy, or reads coming from the reverse strand.

Assemble these reads using one of the approaches discussed, such as greedy shortest common superstring. Since there are many reads, you might consider ways to make the algorithm faster, such as the one discussed in the programming assignment in the previous module.

How many As are there in the full, assembled genome?

Hint: the virus genome you are assembling is exactly 15,894 bases long

✓ Correct

4. How many Ts are there in the full, assembled genome from the previous question?

1 / 1 point

✓ Correct