CS/ECE 374 Spring 2017
Homework 6 Problem 2

Lanxiao Bai (lbai5)
Renheng Ruan (rruan2)
Ho Yin Au (hoyinau2)

Let $X = x_1, x_2, \ldots, x_r$, $Y = y_1, y_2, \ldots, y_s$ and $Z = z_1, z_2, \ldots, z_t$ be three sequences. A common *supersequence* of X, Y and Z is another sequence W such that X, Y and Z are subsequences of W. Suppose $X = a, b, d, c$ and $Y = b, a, b, e, d$ and $Z = b, e, d, c$. A simple common supersequence of X, Y and Z is the concatenation of X, Y and Z which is $a, b, d, c, b, a, b, e, d, b, e, d, c$ and has length 13. A shorter one is b, a, b, e, d, c which has length 6. Describe an efficient algorithm to compute the *length* of the shortest common supersequence of three given sequences X, Y and Z.

Solution: To find the length of shortest common supersequence, we first notice that the searching can be done by recursive algorithm. The naive approach will pick 1 of 3 sequences each time, mark the first element of that sequence as the first element of shortest common supersequence, and recursively search 3 sequences again with the first element of selected sequence removed. However, if the first element of other sequences is the same as the first element of selected sequence, we also need to remove them. After several iteration, we reach the case that 1 sequence is exhausted. We stop considering the exhausted sequence and keep going, we will eventually reach trivial base case where only 1 sequence contains elements. We concatenate those remaining elements to the list of elements we removed during search, we get a shortest common supersequence.

As we are working on length of shortest common supersequence, we do not need to remember the whole sequence. As we must remove one element from 1 of 3 sequences in each recursive call, we just add 1 after each return. Also, we just return the count of remaining elements in the base step instead of concatenation.

After we get the recursive definition, we make the iterative algorithm. We first define the memorization table M to avoid recalculation. Starting from all 3 given sequences are zero length and 2 of 3 given sequences are zero length, we can easily get the value trivially. Then, we figure out the value of entries in M that 1 of 3 given sequences are zero length, and use that information again to find out the final solution.

So, finding length of shortest common supersequence (LSCS) is defined as follow:

```
LSCS(X[1..i], Y[1..j], Z[1..k]):

  INT M[0..i][0..j][0..k]

  for p = 0 to i
      M[p][0][0] = p
  for q = 0 to j
      M[0][q][0] = q
  for r = 0 to k
      M[0][0][r] = r
  for p = 1 to i
      for q = 1 to j
          if X[p] == Y[q]
              M[p][q][0] = 1 + M[p-1][q-1][0]
          else
              M[p][q][0] = 1 + min{M[p-1][q][0], M[p][q-1][0]}
  for q = 1 to j
      for r = 1 to k
          if Y[q] == Z[r]
              M[0][q][r] = 1 + M[0][q-1][r-1]
          else
              M[0][q][r] = 1 + min{M[0][q-1][r], M[0][q][r-1]}
  for p = 1 to i
      for r = 1 to k
          if X[p] == Z[r]
              M[p][0][r] = 1 + M[p-1][0][r-1]
          else
              M[p][0][r] = 1 + min{M[p-1][0][r], M[p][0][r-1]}
  for p = 1 to i
      for q = 1 to j
          for r = 1 to k
              if X[p] == Y[q] and Y[q] == Z[r]
                  M[p][q][r] = 1 + M[p-1][q-1][r-1]
              else if X[p] == Y[q]
                  M[p][q][r] = 1 + min{M[p-1][q-1][r], M[p][q][r-1]}
              else if Y[q] == Z[r]
                  M[p][q][r] = 1 + min{M[p][q-1][r-1], M[p-1][q][r]}
              else if X[p] == Z[r]
                  M[p][q][r] = 1 + min{M[p-1][q][r-1], M[p][q-1][r]}
              else
                  M[p][q][r] = 1 + min{M[p-1][q][r], M[p][q-1][r], M[p][q][r-1]}
  return M[i][j][k]
```

The storage space is $O((i+1)(j+1)(k+1)) = O(ijk)$ as we consider all the prefix of 3 sequences. The running time is also $O((i+1)(j+1)(k+1)) = O(ijk)$ as we only need constant time to fill up each entry of M if we obey the order. ■