Problem 1. Using the recursive formulate, determine the longest common subsequence of *aglet* and *racket* by filling in the chart below (*ϵ* is the empty string as before).

<table>
<thead>
<tr>
<th></th>
<th>ϵ</th>
<th>a</th>
<th>g</th>
<th>l</th>
<th>e</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>ϵ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>k</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Solution 1.

<table>
<thead>
<tr>
<th></th>
<th>ϵ</th>
<th>a</th>
<th>g</th>
<th>l</th>
<th>e</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
</tr>
<tr>
<td>r</td>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
<td>ϵ</td>
</tr>
<tr>
<td>a</td>
<td>ϵ</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>c</td>
<td>ϵ</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>k</td>
<td>ϵ</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>e</td>
<td>ϵ</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>ae</td>
<td>ae</td>
</tr>
<tr>
<td>t</td>
<td>ϵ</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>ae</td>
<td>aet</td>
</tr>
</tbody>
</table>

A *supersequence* of a sequence *A* is any sequence *B* for which *A* is a subsequence of *B*. For example, *amends* is a super sequence of *ads*.

The shortest common supersequence of two sequences *A* and *B* is exactly what it says on the tin. It is the shortest sequence *C* for which both *A* and *B* are subsequences of *C*. For example, the shortest common supersequence of *extra* and *exact* is *exactra*.

Problem 2. Use a technique similar to the previous problem, determine the shortest common supersequence of *aglet* and *racket* by filling in the chart below (*ϵ* is the empty string as before).
### Problem 3.

Give a recursive definition for the shortest common supersequence of two sequences $A = a_1 \ldots a_n$ and $B = b_1 \ldots b_m$. You may give an English description if you wish.

#### Solution 3.

$$SCS(a_1 \ldots a_n, b_1 \ldots b_m) = \begin{cases} \epsilon & n = m = 0 \\ a_1 \ldots a_n & m = 0 \\ b_1 \ldots b_m & n = 0 \\ SCS(a_1 \ldots a_{n-1}, b_1 \ldots b_{m-1})a_n & a_n = b_m \\ shorterof \left( SCS(a_1 \ldots a_{n-1}, b_1 \ldots b_{m-1})a_n \right) & \text{otherwise} \end{cases}$$

### Problem 4.

Give an (efficient) dynamic programming algorithm that constructs the shortest common supersequence of two sequences.

#### Solution 4.
Algorithm 1: SCS(A, B)

Input: Sequences A = a₁...aₙ and B = b₁...bₘ
Output: The shortest common supersequence of A and B
Let \( M: \mathbb{N} \times \mathbb{N} \rightarrow \Sigma^* \) map pairs of subsequence lengths to sequences

Initialize \( M(0, 0) = \epsilon \)
Initialize \( M(i, 0) = a₁...a_i \) for all \( 0 < i \leq n \)
Initialize \( M(0, j) = b₁...b_j \) for all \( 0 < j \leq m \)

for \( i = 1 \) to \( n \) do
  for \( j = 1 \) to \( m \) do
    if \( a_i = b_j \) then
      \( M(i, j) = M(i-1, j-1)a_i \)
    else if \( |M(i-1, j)| \leq |M(i, j-1)| \) then
      \( M(i, j) = M(i-1, j)a_i \)
    else
      \( M(i, j) = M(i, j-1)b_i \)

return \( M(n, m) \)

Problem 5. The longest common subsequence defines an edit distance by counting the minimum number of deletions needed to make both sequences equal to each other (i.e. equal to their longest common subsequence). Given sequences A and B, the edit distance \( d_{LCS}(A, B) \) is

\[
d_{LCS}(A, B) = (|A| - |LCS(A, B)|) + (|B| - |LCS(A, B)|) = |A| + |B| - 2|LCS(A, B)|.
\]

The shortest common supersequence defines an edit distance as well. We need only count the minimum number of insertions required to make both sequences equal to each other (i.e. equal to their shortest common supersequence). Given sequences A and B, define the edit distance \( d_{SCS}(A, B) \) in terms of A, B, and \( SCS(A, B) \).

Solution 5.

\[
d_{SCS}(A, B) = (|SCS(A, B)| - |A|) + (|LCS(A, B)| - |B|) = 2|SCS(A, B)| - |A| - |B|.
\]

Problem 6. Below, we give an intuitive argument to show

\[
|LCS(A, B)| + |SCS(A, B)| = |A| + |B|.
\]

LCS(A, B) is the maximal length sequence of items A and B have in common in a common order. If we remove these elements from A to get \( A' \) and from B to get \( B' \), then \( A' \) and \( B' \) together constitute the elements A and B do not share in a maximally common order. So \( A' \) and \( B' \) are the items A and B do not share in a common order when the maximum number of items in a common order are removed. There are the minimum set of items we must copy/paste, together with LCS(A, B), to get a common supersequence.
of $A$ and $B$. Thus $A'$ and $B'$ together with $LCS(A, B)$ must contain exactly the items in the shortest common supersequence. \hfill \square

Prove algebraically that $d_{LCS}(A, B) = d_{SCS}(A, B)$.

**Solution 6.**

\[
d_{SCS}(A, B) = 2|SCS(A, B)| - |A| - |B|
\]
\[= 2(|A| + |B| - |LCS(A, B)|) - |A| - |B|
\]
\[= |A| + |B| - 2|LCS(A, B)|
\]
\[= d_{LCS}(A, B)
\]
\hfill \square