```
Sequence (E, C): trait
  % Comparison, subsequences
  assumes StrictPartialOrder (>, E)
  includes
    LexicographicOrder,
    String
  introduces
    isPrefix, isSubstring, isSubsequence: C, C → Bool
    find: C, C → Int
  asserts ∀  e, e1, e2: E, s, s1, s2: C
    isPrefix(s1, s2) == s1 = prefix(s2, len(s1));
    isSubstring(s1, s2) ==
      isPrefix(s1, s2) ∨ isSubstring(s1, tail(s2));
    isSubsequence(empty, s);
    ¬isSubsequence(e ⊣ s, empty);
    isSubsequence(e1 ⊣ s1, e2 ⊣ s2) ==
      (e1 = e2 ∧ isSubsequence(s1, s2))
        ∨ isSubsequence(e1 ⊣ s1, s2);
    find(empty, empty) == 0;
    find(s1, e -| s2) ==
      if isPrefix(s1, e -| s2) then 0
      else find(s1, s2) + 1
  implies
    IsPO (isPrefix, C),
    IsPO (isSubstring, C),
    IsPO (isSubsequence, C)
    ∀ s, s1, s2: C, i, n: Int
      isPrefix(prefix(s, n), s);
      isSubstring(substring(s, i, n), s);
      isSubstring(s1, s2) ⇒ isSubsequence(s1, s2)
    converts
        isPrefix, isSubstring, isSubsequence, find
      exempting ∀ s: C, e: E  find(e ⊣ s, empty)
```