# IEEE Standards Interpretations for IEEE Std 1003.2™-1992 IEEE Standard for Information Technology--Portable Operating System Interfaces (POSIX®)--Part 2: Shell and Utilities



**Interpretation Request #40**
**Topic:** I18N issues **Relevant Clauses:** 2.5.2.2

A POSIX defined a mechanism for talking about multi- character sequences as a single unit, namely as collating elements (CEs). Although CEs are motivated by sorting issues, they appear in REs. This obviously leads to the question of how to parse text into CEs?

There are many possible answers, and furthermore, the parsing might be affected by context. For example, given the usual alphabet augmented by the collating element <ij> defined as <i><j>, can the string ij ever be parsed as two collating elements? [1] 2.5.2.2 says in the context of sorting, "strings are first broken up into a series of collating elements" (line 1668). Does this apply to pattern matching? And if so, how exactly is this done (for sorting or pattern matching)?

Proposed suggeston: Add the following text somewhere; this text should be referred to by line 1668 and by the general RE introduction (2.8.2). "When a string is interpreted as a sequence of CEs, the sequence shall be as found by the following process: starting at the first character of the string, determine the longest prefix of the string that matches a CE, add that CE to the sequence and continue this process with the character after that prefix until the string is exhausted."

Note that this applies even if a sort key indicates that a piece of the text is processed in backwards (right- to-left) order; that is, the right-to-left processing applies to the CEs found by a left-to-right lexical scan. This is the greedy algorithm normally done in lexical analysis. Any other choice would require backtracking with potentially exponential run-time. It implies that, when <i><j> is a collating element, under no circumstances can a bracket expression match the i alone in the string ij. In particular, neither [[.i.][.ij.]]j nor

[[.i.]]j matches ij. By contrast, i[[.j.]] does match ij, because in this regular expression i denotes a character and is unaffected by concerns about collating elements.

**Interpretation Response**
The standard is unclear on this issue, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

**Rationale for Interpretation**
None.