recognize them easily, regardless of the grammatical forms (word forms) in which they are used. For example, if we know the word "system," we will not be surprised by the plural form of this word, "systems," or its form "system's." Moreover, having read the text (understood its meaning), we may not remember all the word forms used in it, although they correspond to familiar words. Recognizing words during the automatic indexing procedure is quite a different thing. Many problems arise here. Indeed, a computer does not know any natural language and consequently does not understand the text. It not only does not understand text (the meaning), but it also does not understand the meanings of words. For the computer, a word is not specifically a word, but a certain set of characters (to be precise, codes of characters). Therefore, the word "system" is one set of characters for the computer, "systems" is another, and "system's" is again another distinct set of characters. However, when indexing, we would like the computer to respond to these words in the same manner. Therefore, in automatically comparing words belonging to the text and to the descriptor dictionary, the computer encounters the problem of identifying various forms of the same word.

Traditionally, this problem has been solved in various languages by using the stem (main part) of the word. Researchers have assumed that the same stem corresponds to the same term. In reality this is not so. Words that have the same stem, such as KINDLE and KINDLY (we are not even speaking about homonyms), sometimes should be distinguished. Nevertheless, this idea has appeared to be rather viable. Its implementation normally requires all descriptors to have conditional equivalence classes consisting of lists of stems. These particular lists are then compared with stems separated from words of the text being indexed. If a stem of a word coincides with a stem contained in a certain conditional equivalence class, then this word is included in the document profile. Next we briefly consider the problem of automatic stem separation.

Today, one can find many stemming algorithms in scientific literature, and not only in the English language (Dawson, 1974; Frakes, 1992; Lovins, 1968; Paice, 1990; Porter, 1980; Salton, 1968). Functioning algorithms first appeared in the early 1960s. At that time the main aim of their developers was not automatic indexing, but automatic translation from one natural language to another. Later, the developers of stemming algorithms began to concentrate on automatic indexing, partially due to their modest successes in the automatic translation from natural languages to natural languages. Another cause was that the automatic indexing (translation to artificial languages) was becoming more and more popular.

The majority of the algorithms proposed are aimed mainly at separate affixes and endings of an analyzed word. Precompiled lists of suffixes, prefixes, and endings are used for this purpose. Suffix lists are frequently unified with lists of endings. These lists contain all known suffixes with all possible endings together with separate endings. Researchers frequently use these lists to determine stems of English words. Figure 6.2 gives a portion of such a list compiled by students in the Department of Computer and Information Science at Fordham University during an information retrieval system course.

e	en	ently	est
eable	ence	entum	et
eal	ency	eous	eta
ectual	eness	er	etion
ed	ening	ered	etic
edly	ent	erer	ette
edness	entia	eress	etum
ee	entiae	erial	ety
eer	ential	ery	eur
el	entialness	es	euse
ely	entiate	escent	
ement	entiation	ess	

Figure 6.2
Excerpt from a typical suffix list.

The automatic stemming process consists of the character-to-character comparison of a given word with the list of suffixes. The comparison is performed from right to left. When a suffix or a group of possible suffixes is found in the word, the longest possible suffix is separated. It is clear that strict compliance with this rule will sometimes fail to provide stems (such as will occur when one separates the suffix ING from the word RING or the suffix ANCY from the word FANCY and so forth). Moreover, in some cases no part of the word will remain (for instance, when we try to separate the suffix ANTIC from the word ANTIC or the suffix ARISE from the word ARISE and so forth).

In this connection, van Rijsbergen wrote:

> Unfortunately, context free removal leads to a significant error rate. For example, we may well want UAL removed from FACTUAL but not from EQUAL. To avoid erroneously removing suffixes context rules are devised so that a suffix will be removed only if the context is right. "Right" may mean a number of things: (1) the length of the remaining stem exceeds a given number; the default is usually 2; (2) the stem-ending satisfies a certain condition, e.g. does not end with Q. (van Rijsbergen, 1979)

Obviously, the rules given by van Rijsbergen help to resolve the problem with the word EQUAL, but they do not solve many other linguistic problems. However, in our opinion, he was successful in indicating a certain general