7

Sorting and Searching

For a collection of data objects, we have discussed some data organizing techniques that use array, linked list, stack, queue, tree, and graph objects (to be discussed later). Such basic operations as insertion, deletion, and even searching for these objects were discussed and implemented. A wise selection of one or more such objects for an application is influenced by the efficiency of sorting or searching or both operations. In this chapter, we discuss, implement, and compare several sorting algorithms and several searching algorithms in a sorted (ordered) or unsorted (unordered) data structure. For the purpose of sorting and searching, each data object contains an uniform key (often called a primary key), which is used to perform these operations. The searching based on multiple (primary and secondary) keys will be discussed in Chapter 9.

Using a sorting method, a set of data elements is arranged in ascending or descending order based on the associated keys. Two major types of sorting techniques will be explored: those based on a full comparison of key values and those that exploit the digital properties of the keys. Sorting is useful in such applications as customer names, grades of students, etc. It is also useful to enhance the performance of a search process on an ordered list.

A searching method looks for specified data in an ordered or unordered set of data. When the target data are found in that set, the information of the target data is retrieved for further processing.

7.1 Sorting Methods

In this section, techniques for rearranging data objects in a sorted order will be discussed and implemented. Each data object or element is conveniently viewed as a record containing an identifying key value (numeric or alphabetic). A sort process on a given data set is guided by these keys and is based on one of two fundamental techniques:

- Key comparative sorts, or
- Radix (digital) sorts.

Sorting methods typically involve the two basic operations of comparing two keys and exchanging or swapping two records. Key comparative sorts
follow this model; the sorting criteria (equality with the key) may be based on the numeric or alphabetic (lexicographic) value of the data objects. However, for many applications it is advantageous to exploit the digital properties of the keys. Radix sorts proceed by "comparing" the individual "bits" of a key instead of requiring full key value comparisons.

Sorting methods are categorized into several groups based upon data structures and storage requirements. The primary two categories are (1) internal sorting methods, that is, sorting methods that are applied to the list of data elements small enough to fit in the internal main memory; and (2) external sorting methods, that is, sorting methods that are applied to larger list of data elements residing in an external storage (e.g., files on a disk). The list of discussed and implemented internal sorting methods include insertion sort, selection sort, quick sort, merge sort, tree sort, heap sort, straight and exchange radix sorts, and shell sort. The merge sort as the external sort method is also discussed and implemented.

The fundamental steps in a sort method are:

Step 1. Set exchange = TRUE.

Step 2. Search for a minimum (maximum) data element specified by a given key in a set (collection) of data elements.

Step 3. If the exchange criterion is satisfied, exchange the data element with the other element(s) to bring it in the proper (i.e., sorted) position; and set exchange = TRUE. Otherwise, exchange = FALSE.

Step 4. Repeat steps 2 and 3 until exchange is FALSE, i.e., there are no more exchanges.

Step 5. Arranging the given set of data elements in an ascending or descending order is now complete.

The recursive definition of sorting methods contains:

Step 1. Split or divide the data set into two smaller subsets, say S1 and S2.

Step 2. Recursively sort the smaller set S1 using a sort method.

Step 3. Recursively sort the smaller set S2 using a sort method.

Step 4. Merge or join the two sorted subsets S1' and S2' into the sorted data set S'.

Two groups for the most often used sorting methods are: