**Basic Sorting Algorithms**

* **Sorting** is a process that organizes a collection of data into either
  + **ascending** (non-decreasing) order

or

* + **descending** **order** (non-increasing) order

**Use Cases**

* The need for sorting arises in many situations.
* You may simply want to sort a collection of data before including it in a report.
* Often, however, you must perform a sort as an initialization step for certain algorithms.
* For example, **searching for data** is one of the most common tasks performed by computers.
* When the collection of data to be searched is large, an efficient technique for searching—such as the binary search algorithm—is desirable.
* However, the binary search algorithm requires that the data be sorted. Thus, sorting the data is a step that must precede a binary search on a collection of data that is not already sorted.
* Good sorting algorithms, therefore, are quite valuable.

**Sorting Algorithm Categories**

* You can organize sorting algorithms into two categories.

1. An **internal sort** requires that the collection of data fit entirely in the computer’s main memory.

The algorithms in this chapter are internal sorting algorithms.

1. You use an **external sort** when the collection of data will not fit in the computer’s main memory all at once but must reside in secondary storage, such as on a disk.

Chapter 21 examines external sorts.

**Sort Key**

* It is easy to imagine the results of sorting a collection of primitive data types, such as integers, character, or strings.
* However, consider a collection of objects.
* If each object contains only one data member, sorting the objects is really no different from sorting a collection of integers.
* However, when each object contains several data members, you must know which data member determines the order of the entire object within the collection of data.
* This data member is called the **sort key**.
* For example, if the objects represent people, you might want to sort on their names, their ages, or their zip codes.
* Regardless of your choice of sort key, the sorting algorithm orders entire objects based on only one data member, the sort key.
* For simplicity, the examples in this chapter sort quantities such as numbers or strings.
* All algorithms in this chapter sort the data into ascending order.
* Modifying these algorithms to sort data into descending order is simple.
* Finally, each example assumes that the data resides in an array