

# 17 Advanced Uses of Pointers

*One can only display complex information in the mind.  
Like seeing, movement or flow or alteration of view is more  
important than the static picture, no matter how lovely.*

In previous chapters, we've seen two important uses of pointers. Chapter 11 showed how using a pointer to a variable as a function argument allows the function to modify the variable. Chapter 12 showed how to process arrays by performing arithmetic on pointers to array elements. This chapter completes our coverage of pointers by examining two additional applications: dynamic storage allocation and pointers to functions.

Using dynamic storage allocation, a program can obtain blocks of memory as needed during execution. Section 17.1 explains the basics of dynamic storage allocation. Section 17.2 discusses dynamically allocated strings, which provide more flexibility than ordinary character arrays. Section 17.3 covers dynamic storage allocation for arrays in general. Section 17.4 deals with the issue of storage deallocation—releasing blocks of dynamically allocated memory when they're no longer needed.

Dynamically allocated structures play a big role in C programming, since they can be linked together to form lists, trees, and other highly flexible data structures. Section 17.5 focuses on linked lists, the most fundamental linked data structure. One of the issues that arises in this section—the concept of a “pointer to a pointer”—is important enough to warrant a section of its own (Section 17.6).

Section 17.7 introduces pointers to functions, a surprisingly useful concept. Some of C's most powerful library functions expect function pointers as arguments. We'll examine one of these functions, `qsort`, which is capable of sorting any array.

The last two sections discuss pointer-related features that first appeared in C99: restricted pointers (Section 17.8) and flexible array members (Section 17.9). These features are primarily of interest to advanced C programmers, so both sections can be safely be skipped by the beginner.