

always fits into a single file. Chapter 15 shows how to organize a program that's split over several files.

So far, we've seen that a program may contain the following:

- Preprocessing directives such as `#include` and `#define`
- Type definitions
- Declarations of external variables
- Function prototypes
- Function definitions

C imposes only a few rules on the order of these items: A preprocessing directive doesn't take effect until the line on which it appears. A type name can't be used until it's been defined. A variable can't be used until it's declared. Although C isn't as picky about functions, I strongly recommend that every function be defined or declared prior to its first call. (C99 makes this a requirement anyway.)

C99

There are several ways to organize a program so that these rules are obeyed. Here's one possible ordering:

- `#include` directives
- `#define` directives
- Type definitions
- Declarations of external variables
- Prototypes for functions other than `main`
- Definition of `main`
- Definitions of other functions

It makes sense to put `#include` directives first, since they bring in information that will likely be needed in several places within the program. `#define` directives create macros, which are generally used throughout the program. Putting type definitions above the declarations of external variables is logical, since the declarations of these variables may refer to the type names just defined. Declaring external variables next makes them available to all the functions that follow. Declaring all functions except for `main` avoids the problems that arise when a function is called before the compiler has seen its prototype. This practice also makes it possible to arrange the function definitions in any order whatsoever: alphabetically by function name or with related functions grouped together, for example. Defining `main` before the other functions makes it easier for a reader to locate the program's starting point.

A final suggestion: Precede each function definition by a boxed comment that gives the name of the function, explains its purpose, discusses the meaning of each parameter, describes its return value (if any), and lists any side effects it has (such as modifying external variables).

PROGRAM Classifying a Poker Hand

To show how a C program might be organized, let's attempt a program that's a little more complex than our previous examples. The program will read and classify