Q: **What other incomplete types are there besides incomplete structure types? [p. 492]**

A: One of the most common incomplete types occurs when an array is declared with no specified size:

```
extern int a[];
```

After this declaration (which we first encountered in Section 15.2), a has an incomplete type, because the compiler doesn't know a's length. Presumably a is defined in another file within the program; that definition will supply the missing length. Another incomplete type occurs in declarations that specify no length for an array but provide an initializer:

```
int a[] = {1, 2, 3};
```

In this example, the array a initially has an incomplete type, but the type is completed by the initializer.

C99

flexible array members ➤17.9

Declaring a union tag without specifying the members of the union also creates an incomplete type. Flexible array members (a C99 feature) have an incomplete type. Finally, void is an incomplete type. The void type has the unusual property that it can never be completed, thus making it impossible to declare a variable of this type.

Q: **What other restrictions are there on the use of incomplete types? [p. 492]**

A: The sizeof operator can't be applied to an incomplete type (not surprisingly, since the size of an incomplete type is unknown). A member of a structure or union (other than a flexible array member) can't have an incomplete type. Similarly, the elements of an array can't have an incomplete type. Finally, a parameter in a function definition can't have an incomplete type (although this is allowed in a function *declaration*). The compiler "adjusts" each array parameter in a function definition so that it has a pointer type, thus preventing it from having an incomplete type.

Exercises

Section 19.1

1. A ***queue*** is similar to a stack, except that items are added at one end but removed from the other in a ***FIFO*** (first-in, first-out) fashion. Operations on a queue might include:

    Inserting an item at the end of the queue
    Removing an item from the beginning of the queue
    Returning the first item in the queue (without changing the queue)
    Returning the last item in the queue (without changing the queue)
    Testing whether the queue is empty

    Write an interface for a queue module in the form of a header file named queue.h.

Section 19.2

W 2. Modify the stack2.c file to use the PUBLIC and PRIVATE macros.