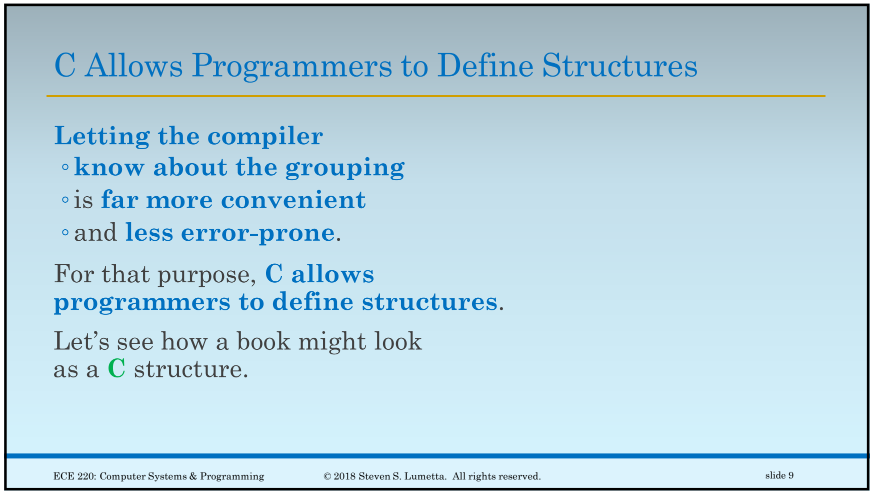


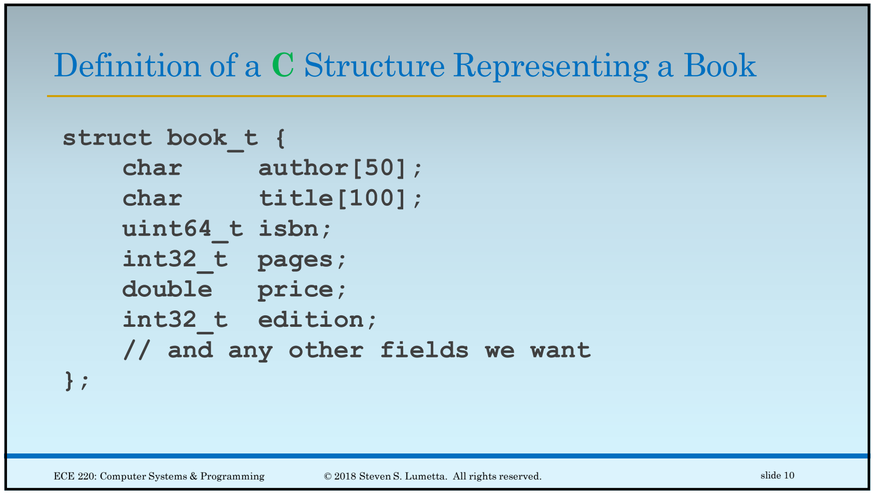
## C Allows Programmers to Define Structures

**Letting the compiler**

- **know about the grouping**
- is **far more convenient**
- and **less error-prone**.

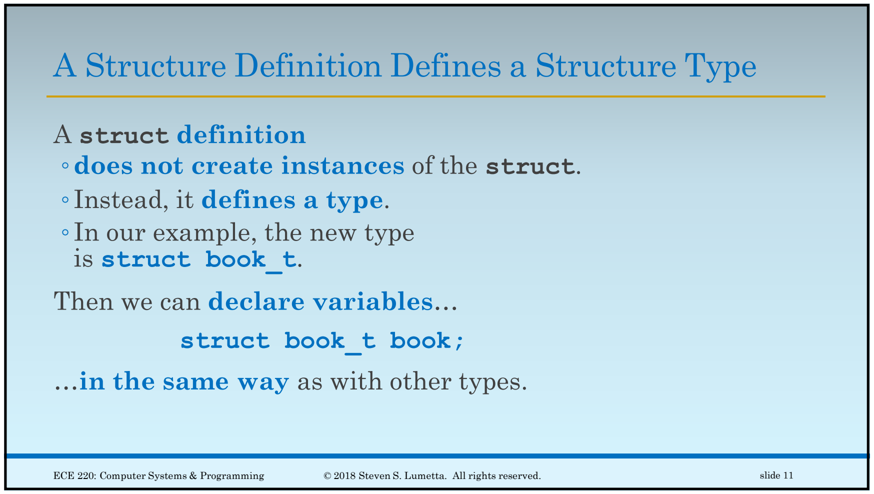
For that purpose, **C allows programmers to define structures**.

Let's see how a book might look as a **C** structure.



## Definition of a C Structure Representing a Book

```
struct book_t {
    char      author[50];
    char      title[100];
    uint64_t isbn;
    int32_t   pages;
    double    price;
    int32_t   edition;
    // and any other fields we want
};
```



## A Structure Definition Defines a Structure Type

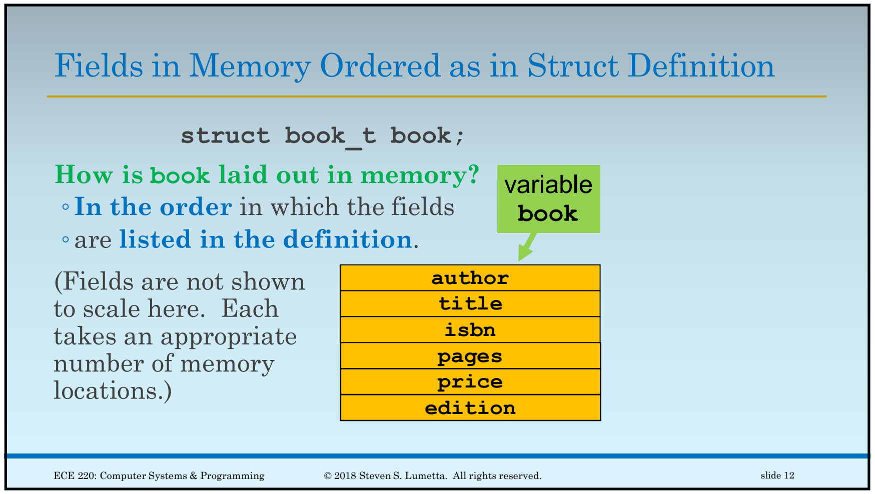
A **`struct` definition**

- **does not create instances** of the **`struct`**.
- Instead, it **defines a type**.
- In our example, the new type is **`struct book_t`**.

Then we can **declare variables**…

```
struct book_t book;
```

…**in the same way** as with other types.



## Fields in Memory Ordered as in Struct Definition

```
struct book_t book;
```

**How is book laid out in memory?**

- **In the order** in which the fields
- are **listed in the definition**.

(Fields are not shown to scale here. Each takes an appropriate number of memory locations.)

variable
book
author
title
isbn
pages
price
edition