Chapter 12 - Structures

At a Glance

Lesson Contents

Chapter Notes

Overview

Chapter 12 covers the use of structures and unions in C. You learn to create and use structures and arrays of structures. You also learn how to pass and return structures to and from functions. You learn how to create and use unions in C. Finally, several common programming and compiler errors are reviewed.

Objectives

- Single structures
- Arrays of structures
- Passing and returning structures
- Unions (optional)
- Common programming and compiler errors

Single Structures

Topic Tip	The Programming Note on page 581 introduces the terms *homogeneous* and *heterogeneous* data structure.

Quick Quiz 1

1. What is the difference between a structure's form and the structure's contents?

2. Assigning actual data values to the data items of a structure is called _____the structure.

3. What is the difference between a homogeneous and a heterogeneous data structure?

4. For non-ANSI C compilers, the keyword_____must be placed before the keyword `struct` for initialization within a local declaration statement.

Arrays of Structures

Topic Tip	Using `typedef` statements with structure declarations is very common. The Programming Note on page 585 provides a good explanation on the use of `typedef` statements.

Quick Quiz 2

1. A_____statement provides a simple method for creating a new and typically shorter name for an existing structure type.

2. What are parallel arrays?

3. What is the problem with parallel arrays?

4. When initializing an array of structures, the_____braces are not necessary.

Unions

Topic Tip	Usually we may have a hard time thinking of uses for unions. You can find a detailed example of a situation that benefits from using a union at **www.cs.utk.edu/~plank/plank/classes/cs140/Spring-1999/notes/Unions/**.

Quick Quiz 3

1. Individual structure members may be passed to a function in the same manner as any _____variable.

2. An alternative to passing a copy of a structure is to pass the_____of the structure.

3. What is a union?

4. How much memory space does a union reserve?

Additional Resources

1. C Tutorial – Lesson 11: Structures:
   http://cplus.about.com/od/beginnerctutorial1/l/aa041602a.htm

2. How C Programming Works: Pointers to Structures:
   http://computer.howstuffworks.com/c31.htm

3. C FAQ: Structures, Unions, and Enumerations:
   http://c-faq.com/struct/

4. C Tutorial: Unions:
   www.sysprog.net/cunions.html

Key Terms

- The structure's **contents内容** consist of the actual data stored in the symbolic names.
- Each of the individual data items in a "structure" (single unit) is an entity by itself that is referred to as a **data field数据段**.
- A structure's **form形式** consists of the symbolic names, data types, and arrangement of individual data fields in the record.
- A record is a **heterogeneous data structure异构数据结构**, which means that each of its components can be of different data types.
- An array is a **homogeneous data structure同质数据结构**, which means that each of its components must be of the same type.
- The data items of a structure are called **members of the structure结构成员**.
- **Parallel arrays并行数组**are two or more arrays, where each array has the same number of elements and the elements in each array are directly related by their position in the arrays.
- Assigning actual data values to the data items of a structure is called **populating the 填充结构**.
- Taken together, all the data fields form a single unit that is referred to as a **record记录**.
- In C, a record is referred to as a **structure结构**, and we use these terms interchangeably.

- When defining structures, if the form of the structure is not followed by any variable names, the list of structure members must be preceded by a user-selected **structure type name**.
- A **union** is a data type that reserves the same area in memory for two or more variables, each of which can be a different data type.