Chapter 11 - Arrays, Addresses, and Pointers

At a Glance

Lesson Contents

Chapter Notes

Overview

Chapter 11 thoroughly covers how pointers work in C. You learn the relationship between arrays and pointers. You also learn how to manipulate pointers and how to pass and use array addresses. You practice processing strings using pointers and learn how to create strings using pointers. Finally, several common programming and compiler errors are reviewed.

Objectives

- Array names as pointers
- Manipulating pointers
- Passing and using array addresses
- Processing strings using pointers
- Creating strings using pointers
- Common programming and compiler errors

Array Names as Pointers

Topic Tip	In C, `5["abcdef"]` is actually valid. For more information, see: **http://c-faq.com/aryptr/joke.html**.

Quick Quiz 1

1. Pointers, both as variables and function parameters, are used to store _____.
2. With respect to pointers, what is an offset?
3. When an array is created, the compiler automatically creates an internal pointer _____.
4. Can a pointer access be replaced using subscript notation? If so, under which circumstances?

Manipulating Pointers

Topic Tip	For a good explanation on C pointer arithmetic, see **www.cs.umd.edu/class/spring2003/cmsc311/Notes/BitOp/pointer.html**.

Quick Quiz 2

1. What is the purpose of adding or subtracting numbers from pointers?
2. When adding or subtracting numbers to pointers, the computer automatically adjusts the number to ensure that the result still "points to" a value of the original _____.
3. How are pointer operations scaled automatically?
4. When initializing pointers you must be careful to set a(n) _____ in the pointer.

Passing and Using Array Addresses

Topic Tip	For more information of pointers to functions, see **www.newty.de/fpt/fpt.html**.

Processing Strings Using Pointers

Topic Tip	It may be useful to see a real compiler implementation of `strcpy()`. For an example, see: **http://freebsd.active-venture.com/FreeBSD-srctree/newsrc/libkern/strcpy.c.html**.

Creating Strings Using Pointers

Topic Tip	It is important that you understand how memory is allocated when a string (character array) is declared versus when a `char` pointer is declared. For a good explanation of the subject, see the Programming Note on page 565.

Quick Quiz 3

1. If `nums` is a two-dimensional integer array, `*(*(nums + 1) + 2)` refers to element _____.

2. What is the main difference between the following declarations?
```
char message1[81] = "this is a string";
char *message2 = "this is a string";
```

3. The header line _____ declares `calc` to be a pointer to a function that returns an integer.

4. What does the declaration `char *seasons[4];` create?

Additional Resources

1. FAQ: Arrays and Pointers:
   http://c-faq.com/~scs/cgi-bin/faqcat.cgi?sec=aryptr

2. Tutorial: Pointers in C and C++:
   http://augustcouncil.com/~tgibson/tutorial/ptr.html

3. The Function Pointer Tutorials:
   www.newty.de/fpt/fpt.html

4. C Tutorial - Lesson 8: An Introduction To Pointers:
   http://cplus.about.com/od/beginnerctutoria1/l/aa040702a.htm

5. How C Programming Works: Pointers:
   http://computer.howstuffworks.com/c20.htm

Key Terms

- **Anagram** 回文构词法 is a rearrangement of letters in a word or phrase that takes another word or phrase.
- One unique feature of pointers is that **offsets** 偏移量 may be included in expressions using pointers.
- A word, phrase, or sentence that reads the same forward and backward, such as *top spot* is a **palindrome**回文.
- When an array is created, the compiler automatically creates an internal **pointer constant** 指针常数 for it and stores the base address of the array in this pointer.