27. Create array declarations for the following problem specifications.
    a. An array to hold the names of five font strings. Initialize the array with your favorites.
    b. An array to hold 12 state names. Initialize with the 12 states closest to your campus.
    c. An array to hold the 10 most common single character middle initials.
    d. An array to store a key for an exam consisting of 15 true/false questions.
    e. Parallel arrays to hold up to 100 checking account check numbers, dates, and check amounts.
28. Explain the difference between the pass by reference and pass by value. When an array is sent to a method, which one is used?

PROGRAMMING EXERCISES

1. Write an application that allows the user to input monthly rainfall amounts for one year storing the values in an array. Create a second array that holds the names of the month. Produce a report showing the month name along with the rainfall amount and its variance from the mean. Calculate and display the average rainfall for the year.
2. Write a program that allows the user to enter any number of names, last name first. Using one of the predefined methods of the `Array class`, order the names in ascending order. Display the results.
3. Create three arrays of type `double`. Do a compile-time initialization and place different values in two of the arrays. Write a program to store the product of the two arrays in the third array. Produce a display using the `MessageBox class` that shows the contents of all three arrays using a single line for an element from all three arrays. Design your solution so that the two original arrays have a different number of elements. Use 1 as the multiplier when you produce the third array.
4. Write an application that can be used to test input values to ensure they fall within an established range. Use an array to keep a count of the number of times each acceptable value was entered. Acceptable values are integers between 0 and 10. Your program should display the total number of valid values inputted as well as the number of invalid entries. Show not only the number of values outside the range, but also the number

of non-numeric invalid values entered. For your final display, output a list of distinct valid entries and a count of how many times that entry occurred. Provide your listing in a tabular format, with values number aligned. Do not display the value unless it was entered.
Use the following test data:

7 2 4 2 q 87 4 6 4 4 7

5. The Ion Realty Sales Corporation would like to have an application showing how each monthly sales contributes to their overall total sales. Write a program that accepts as input any number of monthly sales amounts. After all values have been entered, display a report showing each of the monthly sales amounts and the percentage contribution of the individual monthly sales figure to the overall total sales. Your report should show each original value entered and the percentage that value contributes to the total. You may prompt the user for the number of values to be inputted. Be sure your design is aesthetically pleasing with the percentage rounded to one position and all values number aligned.

6. Write an application that provides statistics about temperatures for a given week. Your solution should be a two-class application that has a one-dimensional array as a data member. The array stores temperatures for any given week. Provide constructors for instantiating the class and methods to return the highest temperature, lowest temperature, average temperature, and the average temperature excluding the lowest temperature. Provide a method that accepts as an argument a temperature and returns the number of days the temperatures were below that value. Override the `ToString( )` method to return a listing of all the temperatures in three-column format and the temperature range for the given week. Write a second class to test your class.

7. Write a two-class application that has as a data member an array that can store state area codes. The class should have a member method that enables users to test an area code to determine if the number is one of the area codes in the state exchange. The member method should use one of the predefined methods of the `Array class` and return `true` if the argument to the method is one of the state codes. Override the `ToString( )` method to return the full list of area codes with each surrounded by parentheses. To test the class, store a list of state codes in a one-dimensional array. Send that array as an argument to the class constructor. Test the instance methods. Your application should work with both an ordered list of area codes or an unordered list.

8. Write a program that accepts any number of homework scores ranging in value from 0 through 10. Prompt the user for a new score if they enter a value outside of the specified range. Prompt the user for a new value if they enter an alphabetic character. Store the values in an array. Calculate the average excluding the lowest and highest scores. Display the average as well as the highest and lowest scores that were discarded.

9. Write a program that allows any number of values between 0 and 10 to be entered. When the user stops entering values, display a frequency distribution bar chart. Use asterisks to show the number of times each value was entered. If a given number is not entered, no asterisks should appear on that line. Your application should display error messages if a value outside the acceptable range is entered or if a non-numeric character is entered.

10. Write a program that will produce a report showing the current and maximum enrollments for a number of classes. Your applications should be designed with two classes. The first class should include data members for the name of the course, current enrollment, and maximum enrollment. Include an instance method that returns the number of students that can still enroll in the course. The `ToString( )` method should return the name of the course, current enrollment, and the number of open slots. In the implementation class, declare parallel arrays and do a compile-time initialization for the name of the course, current enrollment, and maximum enrollment. Also declare an array of class objects in your implementation class. Test your application with the following data:

Class name	Current enrollment	Maximum enrollment
CS150	180	200
CS250	21	30
CS270	9	20
CS300	4	20
CS350	20	20