

ARRAY

1. Write a program to input 10 integer numbers in an array named fmax and determine the maximum value and the index number for the maximum. After displaying the numbers, print these two messages (replacing the underlines with the correct values):

*The maximum value is: \_\_\_\_*

*This is element number \_\_\_\_ in the list of numbers*

2. (Electrical Eng.) Write a program that specifies three one-dimensional arrays named current, resistance, and volts. Each array should be capable of holding 10 elements. Using a for loop, input values for the current and resistance arrays. The entries in the volts array should be the product of the corresponding values in the current and resistance arrays (so  $\text{volts}[i] = \text{current}[i] * \text{resistance}[i]$ ). After all the data has been entered, display the following output, with the appropriate value under each column heading:

**Current**

**Resistance**

**Volts**

3. (Practice) Write a C++ program that adds equivalent elements of the two-dimensional arrays named first and second. Both arrays should have two rows and three columns. For example, element  $\text{first}[1][2]$  of the resulting array should be the sum of  $\text{first}[1][2]$  and  $\text{second}[1][2]$ . The first and second arrays should be initialized as follows:

<i>first</i>			<i>second</i>		
16	18	23	24	52	77
54	91	11	16	19	59

ARRAY

4. (Electrical Eng.) a. An engineer has constructed a two-dimensional array of real numbers with three rows and five columns. This array currently contains test voltages of an amplifier. Write a C++ program that interactively inputs 15 array values, and then determines the total number of voltages in these ranges: less than 60, greater than or equal to 60 and less than 70, greater than or equal to 70 and less than 80, greater than or equal to 80 and less than 90, and greater than or equal to 90.
5. (Statistics) Write a program that includes two functions named `calcavg()` and `variance()`. The `calcavg()` function should calculate and return the average of values stored in an array named `testvals`. The array should be declared in `main()` and include the values 89, 95, 72, 83, 99, 54, 86, 75, 92, 73, 79, 75, 82, and 73. The `variance()` function should calculate and return the variance of the data. The variance is obtained by subtracting the average from each value in `testvals`, squaring the values obtained, adding them, and dividing by the number of elements in `testvals`. The values returned from `calcavg()` and `variance()` should be displayed by using `cout` statements in `main()`.
6. (Numerical) Write and test a function that returns the position of the largest and smallest values in an array of double-precision numbers.
7. (Sorting) Read a set of numerical grades from the keyboard into an array. The maximum number of grades to be entered is 50, and data entry should be terminated when a negative number is entered. Have your program sort and print the grades in descending order.
8. (Numerical) a. Define an array with a maximum of 20 integer values, and fill the array with numbers input from the keyboard or assigned by the program. Then

# ARRAY

write a function named `split()` that reads the array and places all zeros or positive numbers in an array named `positive` and all negative numbers in an array named `negative`. Finally, have your program call a function that displays the values in both the `positive` and `negative` arrays.

b. Extend the program written for Exercise 6a to sort the `positive` and `negative` arrays into ascending order before they are displayed.

9. Write a program that reads in an array of type `int`. You may assume that there are fewer than 50 entries in the array. Your program determines how many entries are used. The output is to be a two-column list. The first column is a list of the distinct array elements; the second column is the count of the number of occurrences of each element. The list should be sorted on entries in the first column, largest to smallest.

For the array values:

-12 3 -12 4 1 1 -12 1 -1 1 2 3 4 2 3 -12

the output should be

<u>N</u>	<u>Count</u>
4	2
3	3
2	2
1	4
-1	1
-12	4

ARRAY

10. Write a C++ function, `lastLargestIndex`, that takes as parameters an int array and its size and returns the index of the last occurrence of the largest element in the array. Also, write a program to test your function.
11. Write a C++ function, `smallestIndex`, that takes as parameters an int array and its size and returns the index of the first occurrence of the smallest element in the array. Also, write a program to test your function.
12. Write a function, `insertAt`, that takes four parameters: an array of integers, the number of elements in the array, an integer (say, `insertItem`), and an integer (say, `index`). The function should insert `insertItem` in the array at the position specified by `index`. If `index` is out of range, output an appropriate message. (Note that `index` must be between 0 and the number of elements in the array; that is,  $0 \leq \text{index} < \text{the number of elements in the array}$ .) Assume that the array is unsorted.
13. Write a function, `removeAt`, that takes three parameters: an array of integers, the number of elements in the array, and an integer (say, `index`). The function should delete the array element indicated by `index`. If `index` is out of range or the array is empty, output an appropriate message. (Note that after deleting the element, the number of elements in the array is reduced by 1.) Assume that the array is unsorted.
14. Write a function, `remove`, that takes three parameters: an array of integers, the number of elements in the array, and an integer (say, `removeItem`). The function should find and delete the first occurrence of `removeItem` in the array. If the value does not exist or the array is empty, output an appropriate message. (Note that after deleting the element, the number of elements in the array is reduced by 1.) Assume that the array is unsorted.