5. Write a program that reads in a list of integers into an array with base type *int*. Provide the facility to either read this array from the keyboard or from a file, at the user's option. If the user chooses file input, the program should request a file name. You may assume that there are fewer than 50 entries in the array. Your program determines how many entries there are. 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 example, for the input

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

   the output should be

   ```
    N    Count
    4    2
    3    3
    2    2
    1    4
   -1    1
   -12   4
   ```

12. Write a program that accepts input like the program in Display 7.8 and that outputs a bar graph like the one in that display except that your program will output the bars vertically rather than horizontally. A two-dimensional array may be useful.

DISPLAY 7.8 **Production Graph Program** *(part 1 of 3)*

```
1   //Reads data and displays a bar graph showing productivity for each plant.
2   #include <iostream>
3   #include <cmath>
4   const int NUMBER_OF_PLANTS = 4;

5   void input_data(int a[], int last_plant_number);
6   //Precondition: last_plant_number is the declared size of the array a.
7   //Postcondition: For plant_number = 1 through last_plant_number:
8   //a[plant_number - 1] equals the total production for plant number plant_number.

9   void scale(int a[], int size);
10  //Precondition: a[0] through a[size - 1] each has a nonnegative value.
11  //Postcondition: a[i] has been changed to the number of 1000s (rounded to
12  //an integer) that were originally in a[i], for all i such that 0 <= i <= size - 1.

13  void graph(const int asterisk_count[], int last_plant_number);
14  //Precondition: asterisk_count[0] through asterisk_count[last_plant_number - 1]
15  //have nonnegative values.
16  //Postcondition: A bar graph has been displayed saying that plant
17  //number N has produced asterisk_count[N - 1] 1000s of units, for each N such that
18  //1 <= N <= last_plant_number
```

(continued)

DISPLAY 7.8 Production Graph Program *(part 2 of 3)*

```
19    void get_total(int& sum);
20    //Reads nonnegative integers from the keyboard and
21    //places their total in sum.
22    int round(double number);
23    //Precondition: number >= 0.
24    //Returns number rounded to the nearest integer.
25    void print_asterisks(int n);
26    //Prints n asterisks to the screen.
27    int main( )
28    {
29        using namespace std;
30        int production[NUMBER_OF_PLANTS];
31        cout << "This program displays a graph showing\n"
32             << "production for each plant in the company.\n";
33        input_data(production, NUMBER_OF_PLANTS);
34        scale(production, NUMBER_OF_PLANTS);
35        graph(production, NUMBER_OF_PLANTS);
36        return 0;
37    }
38    //Uses iostream:
39    void input_data(int a[], int last_plant_number)
    <The rest of the definition of input_data is given in Display 7.6.>
40    //Uses iostream:
41    void get_total(int& sum)
    <The rest of the definition of get_total is given in Display 7.6.>
42    void scale(int a[], int size)
    <The rest of the definition of scale is given in Display 7.7.>
43    //Uses cmath:
44    int round(double number)
    <The rest of the definition of round is given in Display 7.7.>
45    //Uses iostream:
46    void graph(const int asterisk_count[], int last_plant_number)
47    {
48        using namespace std;
49        cout << "\nUnits produced in thousands of units:\n";
50        for (int plant_number = 1;
51             plant_number <= last_plant_number; plant_number++)
52        {
53            cout << "Plant #" << plant_number << " ";
54            print_asterisks(asterisk_count[plant_number - 1]);
55            cout << endl;
56        }
57    }
```

(continued)

DISPLAY 7.8 Production Graph Program *(part 3 of 3)*

```
58    //Uses iostream:
59    void print_asterisks(int n)
60    {
61        using namespace std;
62        for (int count = 1; count <= n; count++)
63            cout << "*";
64    }
```