The term \{94, 98, 87, 103, 101\} initializes the first row, represented by \texttt{maxtemps[0]}. As a matter of style, placing each row of data on its own line, if possible, makes the data easier to read.

Listing 5.19 incorporates an initialized two-dimensional array and a nested loop into a program. This time the program reverses the order of the loops, placing the column loop (city index) on the outside and the row loop (year index) on the inside. Also, it uses a common C++ practice of initializing an array of pointers to a set of string constants. That is, \texttt{cities} is declared as an array of pointers-to-\texttt{char}. That makes each element, such as \texttt{cities[0]}, a pointer-to-\texttt{char} that can be initialized to the address of a string. The program initializes \texttt{cities[0]} to the address of the "Gribble City" string, and so on. Thus, this array of pointers behaves like an array of strings.

\begin{verbatim}
Listing 5.19 nested.cpp

// nested.cpp -- nested loops and 2-D array
#include <iostream>
using namespace std;
const int Cities = 5;
const int Years = 4;
int main()
{
    const char * cities[Cities] =   // array of pointers
    {                                // to 5 strings
        "Gribble City",
        "Gribbletown",
        "New Gribble",
        "San Gribble",
        "Gribble Vista"
    };

\end{verbatim}