1. **Ensure that you're in the `learn` folder.**

   Heed the steps in the section "Finding your `learn` directory or folder," earlier in this appendix.

2. **Use your text editor to create your source code file.**

   Use vi, ee, or whatever your favorite text editor is to create and save the source code file. For an example, you can refer to the listing of the GOODBYE.C source in Chapter 1; type that text into your editor.

3. **Compile and link the source code.**

   Compiling and linking are both handled by the GCC command. As an example, here's what you need to type to compile and link the GOODBYE.C source code created in Step 1:

   ```
   gcc goodbye.c -o goodbye
   ```

   The code has four items:

   - `gcc`, the command to compile and link the source code
   - `goodbye.c`, the name of the source code file
   - `-o`, the output switch
   - `goodbye`, the name of the final program

   If you leave off the `-o` switch and its option, GCC creates the program file named `a.out`. I don't recommend this. Instead, remember the `-o` option and specify a name for the output program. The name can be the same as the source code file, but without the `.c` extension.

4. **Run the program.**

   Alas, your operating system doesn't run your program if you type its name at the prompt. That's because Unix runs only programs found on the path, and I don't recommend putting your `learn` directory on the path. (If you create your own programs that you want to run, copy them to a `bin` directory beneath your home directory, and put *that* directory on the path.)

   To get the operating system to notice your program, you have to be specific about where the program lives (in the current folder, for example). You do that by prefixing `./` to the program's name. To run the `goodbye` program, type the following at the prompt:

   ```
   ./goodbye
   ```

   And the program runs.

Those steps are the basic ones you take (all in the `learn` folder) to create the program examples in this book. As I have said, it eventually becomes second nature to you.