return 3.142. *Hint:* Multiply x by 10^n, round to the nearest integer, then divide by 10^n. Be sure that your function works correctly for both positive and negative values of x.

Section 23.4

2. (C99) Write the following function:

   ```
   double evaluate_polynomial(double a[], int n, double x);
   ```

   The function should return the value of the polynomial $a_n x^n + a_{n-1} x^{n-1} + \ldots + a_0$, where the a_i's are stored in corresponding elements of the array `a`, which has length `n` + 1. Have the function use Horner's Rule to compute the value of the polynomial:

   $((\ldots((a_n x + a_{n-1})x + a_{n-2})x + \ldots)x + a_1)x + a_0$

   Use the `fma` function to perform the multiplications and additions.

3. (C99) Check the documentation for your compiler to see if it performs contraction on arithmetic expressions and, if so, under what circumstances.

Section 23.5

4. Using `isalpha` and `isalnum`, write a function that checks whether a string has the syntax of a C identifier (it consists of letters, digits, and underscores, with a letter or underscore at the beginning).

5. Using `isxdigit`, write a function that checks whether a string represents a valid hexadecimal number (it consists solely of hexadecimal digits). If so, the function returns the value of the number as a `long int`. Otherwise, the function returns −1.

Section 23.6

Ⓦ 6. In each of the following cases, indicate which function would be the best to use: `memcpy`, `memmove`, `strcpy`, or `strncpy`. Assume that the indicated action is to be performed by a single function call.

   (a) Moving all elements of an array "down" one position in order to leave room for a new element in position 0.

   (b) Deleting the first character in a null-terminated string by moving all other characters back one position.

   (c) Copying a string into a character array that may not be large enough to hold it. If the array is too small, assume that the string is to be truncated; no null character is necessary at the end.

   (d) Copying the contents of one array variable into another.

7. Section 23.6 explains how to call `strchr` repeatedly to locate all occurrences of a character within a string. Is it possible to locate all occurrences *in reverse order* by calling `strrchr` repeatedly?

Ⓦ 8. Use `strchr` to write the following function:

   ```
   int numchar(const char *s, char ch);
   ```

   `numchar` returns the number of times the character `ch` occurs in the string `s`.

9. Replace the test condition in the following `if` statement by a single call of `strchr`:

   ```
   if (ch == 'a' || ch == 'b' || ch == 'c') ...
   ```

Ⓦ 10. Replace the test condition in the following `if` statement by a single call of `strstr`:

   ```
   if (strcmp(str, "foo") == 0 || strcmp(str, "bar") == 0 ||
       strcmp(str, "baz") == 0) ...
   ```

   Hint: Combine the string literals into a single string, separating them with a special character. Does your solution assume anything about the contents of `str`?