clearerr isn't needed often, since some of the other library functions clear one or both indicators as a side effect.

Q&A

feof
ferror

Q&A

We can call the feof and ferror functions to test a stream's indicators to determine why a prior operation on the stream failed. The call feof(fp) returns a nonzero value if the end-of-file indicator is set for the stream associated with fp. The call ferror(fp) returns a nonzero value if the error indicator is set. Both functions return zero otherwise.

When scanf returns a smaller-than-expected value, we can use feof and ferror to determine the reason. If feof returns a nonzero value, we've reached the end of the input file. If ferror returns a nonzero value, a read error occurred during input. If neither returns a nonzero value, a matching failure must have occurred. Regardless of what the problem was, the return value of scanf tells us how many data items were read before the problem occurred.

To see how feof and ferror might be used, let's write a function that searches a file for a line that begins with an integer. Here's how we intend to call the function:

```
n = find_int("foo");
```

"foo" is the name of the file to be searched. The function returns the value of the integer that it finds, which is then assigned to n. If a problem arises—the file can't be opened, a read error occurs, or no line begins with an integer—find_int will return an error code (–1, –2, or –3, respectively). I'll assume that no line in the file begins with a negative integer.

```
int find_int(const char *filename)
{
  FILE *fp = fopen(filename, "r");
  int n;

  if (fp == NULL)
    return -1;                  /* can't open file */

  while (fscanf(fp, "%d", &n) != 1) {
    if (ferror(fp)) {
      fclose(fp);
      return -2;                /* read error */
    }
    if (feof(fp)) {
      fclose(fp);
      return -3;                /* integer not found */
    }
    fscanf(fp, "%*[^\n]");      /* skips rest of line */
  }

  fclose(fp);
  return n;
}
```

The while loop's controlling expression calls fscanf in an attempt to read an integer from the file. If the attempt fails (fscanf returns a value other than 1),