#include "buffer.h"

/* the buffer */
buffer_item buffer[BUFFER_SIZE];

int insert_item(buffer_item item) {
    /* insert item into buffer
     * return 0 if successful, otherwise
     * return -1 indicating an error condition */
}

int remove_item(buffer_item *item) {
    /* remove an object from buffer
     * placing it in item
     * return 0 if successful, otherwise
     * return -1 indicating an error condition */
}

Figure 6.27 A skeleton program.

The insert_item() and remove_item() functions will synchronize
the producer and consumer using the algorithms outlined in Figures
6.10 and 6.11. The buffer will also require an initialization function that
initializes the mutual-exclusion object mutex along with the empty and
full semaphores.

The main() function will initialize the buffer and create the separate
producer and consumer threads. Once it has created the producer
and consumer threads, the main() function will sleep for a period of
time and, upon awakening, will terminate the application. The main()
function will be passed three parameters on the command line:

a. How long to sleep before terminating
b. The number of producer threads
c. The number of consumer threads

A skeleton for this function appears in Figure 6.28.

#include "buffer.h"

int main(int argc, char *argv[]) {
    /* 2. Initialize buffer */
    /* 3. Create producer thread(s) */
    /* 4. Create consumer thread(s) */
    /* 5. Sleep */
    /* 6. Exit */
}