

```

typeT buf[n];          /* an array of some type T */
int front = 0, rear = 0;
sem empty = n, full = 0;    /* n-2 <= empty+full <= n */
sem mutexD = 1, mutexF = 1; /* for mutual exclusion */
process Producer[i = 1 to M] {
    while (true) {
        ...
        produce message data and deposit it in the buffer;
        P(empty);
        P(mutexD);
        buf[rear] = data; rear = (rear+1) % n;
        V(mutexD);
        V(full);
    }
}
process Consumer[j = 1 to N] {
    while (true) {
        fetch message result and consume it;
        P(full);
        P(mutexF);
        result = buf[front]; front = (front+1) % n;
        V(mutexF);
        V(empty);
        ...
    }
}

```

Figure 4.5 Multiple producers and consumers using semaphores.