

```

1  class Queue<T>
2  {
3      int head, tail;
4      T[] call;
5      public Queue(int capacity)
6      {
7          call = new T[capacity];
8          head = tail = 0;
9      }
10     public void Enq(T x)
11     {
12         Monitor.Enter(this);
13         try
14         {
15             while (tail - head == call.Length)
16             {
17                 Monitor.Wait(this); // queue is full
18             }
19             calls[(tail++) % call.Length] = x;
20             Monitor.Pulse(this); // notify waiting dequeuers
21         }
22         finally
23         {
24             Monitor.Exit(this);
25         }
26     }
27     public T Deq()
28     {
29         Monitor.Enter(this);
30         try
31         {
32             while (tail == head)
33             {
34                 Monitor.Wait(this); // queue is empty
35             }
36             T y = calls[(head++) % call.Length];
37             Monitor.Pulse(this); // notify waiting enqueueers
38             return y;
39         }
40         finally
41         {
42             Monitor.Exit(this);
43         }
44     }
45 }

```

FIGURE A.11 A bounded Queue class.