

```

1  public class UnboundedDeque {
2      private final static int LOG_CAPACITY = 4;
3      private volatile CircularArray tasks;
4      volatile int bottom;
5      AtomicReference<Integer> top;
6      public UnboundedDeque(int logCapacity) {
7          tasks = new CircularArray(logCapacity);
8          top = new AtomicReference<Integer>(0);
9          bottom = 0;
10     }
11     boolean isEmpty() {
12         int localTop = top.get();
13         int localBottom = bottom;
14         return (localBottom <= localTop);
15     }
16
17     public void pushBottom(RecursiveAction r) {
18         int oldBottom = bottom;
19         int oldTop = top.get();
20         CircularArray currentTasks = tasks;
21         int size = oldBottom - oldTop;
22         if (size >= currentTasks.capacity()-1) {
23             currentTasks = currentTasks.resize(oldBottom, oldTop);
24             tasks = currentTasks;
25         }
26         currentTasks.put(oldBottom, r);
27         bottom = oldBottom + 1;
28     }

```

FIGURE 16.14 The UnboundedDeque class: fields, constructor, pushBottom(), and isEmpty() methods.