ALGORITHM 1.4  Bag

```java
import java.util.Iterator;

public class Bag<Item> implements Iterable<Item> {
    private Node first;  // first node in list

    private class Node {
        Item item;
        Node next;
    }

    public void add(Item item) {
        // same as push() in Stack
        Node oldfirst = first;
        first = new Node();
        first.item = item;
        first.next = oldfirst;
    }

    public Iterator<Item> iterator() {
        return new ListIterator();
    }

    private class ListIterator implements Iterator<Item> {
        private Node current = first;

        public boolean hasNext() {
            return current != null;
        }

        public void remove() {
        }

        public Item next() {
            Item item = current.item;
            current = current.next;
            return item;
        }
    }
}
```

This Bag implementation maintains a linked list of the items provided in calls to add(). Code for isEmpty() and size() is the same as in Stack and is omitted. The iterator traverses the list, maintaining the current node in current. We can make Stack and Queue iterable by adding the code highlighted in red to ALGORITHMS 1.1 and 1.2, because they use the same underlying data structure and Stack and Queue maintain the list in LIFO and FIFO order, respectively.