```
1   // Multiple correct solutions accepted.
2
3
4   // COPY CONSTRUCTOR (the reverse)
5   template<typename ItemType>
6   LinkedBag<ItemType>::LinkedBag(const LinkedBag<ItemType>& aBag, const string&
    mode) {
7
8       itemCount = aBag.itemCount;
9       Node<ItemType>* origChainPtr = aBag.headPtr;
10
11      if (origChainPtr == nullptr) {
12          headPtr = nullptr;
13      }
14      else {
15          headPtr = new Node<ItemType>(origChainPtr->getItem());
16          origChainPtr = origChainPtr->getNext();
17
18          while (origChainPtr != nullptr)
19          {
20              Node<ItemType>* newNodePtr = new
                Node<ItemType>(origChainPtr->getItem(), headPtr);
21              headPtr = newNodePtr;
22              origChainPtr = origChainPtr->getNext();
23          }
24      }
25  } // end COPY CONSTRUCTOR (the reverse)
26
27  /* OUTPUT
28
29  !Display bag : e l e c t r i c a l
30  -----------> 10 item(s) total
31
32  !Display bag : l a c i r t c e l e
33  -----------> 10 item(s) total
34
35  */
36
37
38
39  // removeLastThree
40  // Assumption: It is OK to use either the head item or the tail item
41  // to replace the item which we need to remove.
42  template<typename ItemType>
43  bool LinkedBag<ItemType>::removeLastThree(const ItemType& itemToRemove)
44  {
45      const int totalToRemove = 3;
46      int totalRemoved = 0;
47
48      LinkedBag<ItemType>* revBag{ new LinkedBag(*this, "reverse") };
49      Node<ItemType>* curPtr{ revBag->headPtr };
50
51      while (curPtr != nullptr && totalRemoved != totalToRemove)
52      {
53          if (itemToRemove != curPtr->getItem()) {
54              curPtr = curPtr->getNext();
55          }
56          else {
57              curPtr->setItem(revBag->headPtr->getItem());
58              Node<ItemType>* nodeToDelete{ revBag->headPtr };
59
60              revBag->headPtr = revBag->headPtr->getNext();
61              curPtr = curPtr->getNext();
62
63              nodeToDelete->setNext(nullptr);
```

```
            delete nodeToDelete;
            nodeToDelete = nullptr;

            totalRemoved++;
            itemCount--;
        }
    }

    LinkedBag<ItemType>* retBag{ new LinkedBag(*revBag, "reverse") };
    this->headPtr = retBag->headPtr;

    return true;
} // end removeLastThree

/* OUTPUT

----- removeLastThree TEST -----

 !Display bag: r e e n i g n e l a c i r t c e l e
  -----------> 18 item(s) total

 !Display bag: r e e n i g n l l a c i r t c
  -----------> 15 item(s) total

*/
```