```cpp
/*
    Name header
*/

#include "DoublyList.h"

// createAList
void DoublyList::createAList()
{
    /*
        NOTE:

            * You will need to declare one pointer and
              you may re-use this pointer throughout the function, but
              you are NOT allowed to create additional pointers.

            * DO NOT REMOVE EXISTING COMMENTS.

            * Pay CLOSE attention to instructions.
    */

    /*-----------------------------------------------------------------
    SECTION 1
    -----------------------------------------------------------------*/

    // Create a node that stores the value 2 and make
    // this node be the first node of the calling object.
    // List becomes: 2
    // Use the overloaded constructor.

    first = new Node(2, nullptr, nullptr);
    last = first;

    // Update count;
    ++count;

    cout << "Section 1 - TEST ALL" << endl;
    testAll();

    /*-----------------------------------------------------------------
    SECTION 2
    -----------------------------------------------------------------*/

    // Create another node that stores the value 3 and
    // insert this node to the left of the node that is
    // storing value 2.
    // List becomes: 3 2

    first->setPrev(new Node(3, nullptr, first));
    first = first->getPrev();

    // Update count;
```

```
53      ++count;
54
55      cout << "\n\nSection 2 - TEST ALL" << endl;
56      testAll();
57
58      /*-------------------------------------------------------------
59      SECTION 3
60      -------------------------------------------------------------*/
61
62      // Create another node that stores the value 4 and
63      // insert this node to the right of the node that is
64      // storing value 3.
65      // List becomes: 3 4 2
66      // NO MORE than 3 statements.
67
68      Node *pNode = new Node(4, first, last);
69      first->setNext(pNode);
70      last->setPrev(pNode);
71
72      // Update count;
73      ++count;
74
75      cout << "\n\nSection 3 - TEST ALL" << endl;
76      testAll();
77
78      /*-------------------------------------------------------------
79      SECTION 4
80      -------------------------------------------------------------*/
81
82      // Delete the first node.
83      // List becomes: 4 2
84
85      pNode = first;
86      first = first->getNext();
87      first->setPrev(nullptr);
88      delete pNode;
89      pNode = nullptr;
90
91      // Update count.
92      --count;
93
94      cout << "\n\nSection 4 - TEST ALL" << endl;
95      testAll();
96
97      /*-------------------------------------------------------------
98      SECTION 5
99      -------------------------------------------------------------*/
100
101     // Insert three nodes at the end of the list storing
102     // 5 6 7 in this order.
103     // List becomes: 4 2 5 6 7
104
```

```
105     last->setNext(new Node(5, last, nullptr));
106     last = last->getNext();
107     last->setNext(new Node(6, last, nullptr));
108     last = last->getNext();
109     last->setNext(new Node(7, last, nullptr));
110     last = last->getNext();
111
112     // Update count.
113     // One statement only.
114     count = 5;
115
116     cout << "\n\nSection 5 - TEST ALL" << endl;
117     testAll();
118
119     /*-----------------------------------------------------------------
120     SECTION 6
121     -----------------------------------------------------------------*/
122
123     // Move last node to second position.
124     // Here steps are very important. Carefully think
125     // how you can move nodes around without losing any
126     // nodes and keeping all pointers pointing to the
127     // correct nodes.
128     // Note:
129     //      You may NOT create an additional node.
130     //      NO loops are necessary.
131     // List is 4 2 5 6 7 => will become 4 7 2 5 6
132
133     last = last->getPrev();
134     first->getNext()->setPrev(last->getNext());
135     last->getNext()->setNext(first->getNext());
136     first->setNext(last->getNext());
137     last->getNext()->setPrev(first);
138     last->setNext(nullptr);
139
140     cout << "\n\nSection 6 - TEST ALL" << endl;
141     testAll();
142
143     /*-----------------------------------------------------------------
144     SECTION 7
145     -----------------------------------------------------------------*/
146
147     // Move the first node in between the node before last and
148     // last node (the second node will become the first node
149     // in the list, and the first node will become the before-last
150     // node in the list).
151     //      You may NOT create an additional node.
152     //      No loops are necessary.
153     // List is 4 7 2 5 6 => will become 7 2 5 4 6
154
155     first = first->getNext();
156     last->getPrev()->setNext(first->getPrev());
```

```
157      first->getPrev()->setPrev(last->getPrev());
158      last->setPrev(first->getPrev());
159      first->getPrev()->setNext(last);
160      first->setPrev(nullptr);
161
162      cout << "\n\nSection 7 - TEST ALL" << endl;
163      testAll();
164
165      /*-----------------------------------------------------------------
166      SECTION 8
167      -----------------------------------------------------------------*/
168
169      // WITHOUT moving any nodes, swap around the values to
170      // create an ordered list.
171      // Note that there is no need to move the value 5.
172      // You may declare an int, BUT do NOT use any literals.
173      // List will become: 2 4 5 6 7
174
175      int temp = first->getData();
176      first->setData(first->getNext()->getData());
177      first->getNext()->setData(last->getPrev()->getData());
178      last->getPrev()->setData(last->getData());
179      last->setData(temp);
180
181      cout << "\n\nSection 8 - TEST ALL" << endl;
182      testAll();
183
184      /*-----------------------------------------------------------------
185      SECTION 9
186      -----------------------------------------------------------------*/
187
188      // Add two nodes storing 1 and 3 to complete the ordered list.
189      // List becomes: 1 2 3 4 5 6 7
190
191      pNode = new Node(3, first, first->getNext()); // easier to insert 3 first
192      first->getNext()->setPrev(pNode);
193      first->setNext(pNode);
194
195      first->setPrev(new Node(1, nullptr, first));
196      first = first->getPrev();
197
198      // Add 2 to count.
199      count += 2;
200
201      cout << "\n\nSection 9 - TEST ALL" << endl;
202      testAll();
203
204  }
```