```cpp
/*
    Salcedo, Salvador

    CS A250
    March 15, 2019

    Lab 7
*/

#include "DoublyList.h"

DoublyList::DoublyList()
{
    first = nullptr;
    last = nullptr;
    count = 0;
}

void DoublyList::insertBack(int newData) {
    Node *newNode = new Node(newData, last, nullptr);
    if (first == nullptr) {
        first = newNode;
        last = newNode;
    }
    else {
        last->setNext(newNode);
        last = last->getNext();
    }
    count++;
}

bool DoublyList::search(int searchData) const
{
    Node *current = first;
    while (current != nullptr) {
        if (current->getData() == searchData)
            return true;
        else
            current = current->getNext();
    }
    return false;
}

void DoublyList::deleteNode(int deleteData) {
    if (first == nullptr) {
        cerr << "Cannot delete from an empty list." << endl;
    }
    else {
        Node *current = first;
```

```
50          if (current->getData() == deleteData) {
51              first = first->getNext();
52
53              if (first == nullptr)
54                  last = nullptr;
55              else
56                  first->setPrev(nullptr);
57
58              delete current;
59              current = nullptr;
60              --count;
61          }
62          else {
63              bool found = false;
64              while (current != nullptr && !found) {
65                  if (current->getData() == deleteData)
66                      found = true;
67                  else
68                      current = current->getNext();
69              }
70              if (current == nullptr)
71                  cerr << "The item to be deleted is not in the list." << endl;
72              else {
73                  if (current != last) {
74                      current->getPrev()->setNext(current->getNext());
75                      current->getNext()->setPrev(current->getPrev());
76                  }
77                  else {
78                      last = current->getPrev();
79                      last->setNext(nullptr);
80                  }
81                  --count;
82                  delete current;
83                  current = nullptr;
84              }
85          }
86      }
87  }
88
89  void DoublyList::print() const {
90      if (count == 0)
91          cerr << "List is empty. Cannot print." << endl;
92      else {
93          Node *temp = first;
94
95          while (temp != nullptr) {
96              cout << temp->getData() << " ";
97              temp = temp->getNext();
98          }
```

```
 99          cout << endl;
100      }
101  }
102
103  void DoublyList::reversePrint() const {
104      if (count == 0)
105          cerr << "List is empty. Cannot print." << endl;
106      else {
107          Node *temp = last;
108          while (temp != nullptr) {
109              cout << temp->getData() << " ";
110              temp = temp->getPrev();
111          }
112          cout << endl;
113      }
114  }
115
116  void DoublyList::destroyList() {
117      Node *temp = first;
118      while (temp != nullptr) {
119          first = first->getNext();
120          delete temp;
121          temp = first;
122      }
123      count = 0;
124      last = nullptr;
125  }
126
127  DoublyList::~DoublyList() {
128      destroyList();
129  }
```