Russ Johnson
Data Structures and Algorithms
Assignment 4
January 28, 2013

```
> g++ -Wall -g A.cpp -o A
> ./A
Title: Die Hard
Minutes: 130
Price: 9.99
Title: Stein's Gate
Minutes: 120
Price: 10.11
Title: Bunraku
Minutes: 121
Price: 3.56
Title: Die Hard
Minutes: 130
Price: 9.99
Title: Stein's Gate
Minutes: 120
Price: 1.11
Title: Bunraku
Minutes: 121
Price: 0
Die Hard
120
0
```

```
//dvd.h
#ifndef DVD_H
#define DVD_H

#include <string>
using namespace std;


// class definition
class DVD {
    const string title;
    const int minutes;
    double price;

  public:
    DVD(const string &t, int m, double p);
    void output() const;
    void changeprice(double p_new);
    const string & get_title() const {return title;}
    const int get_minutes() const {return minutes;}
    const double get_price() const {return price;}
  };

#endif  /* DVD_H */
```

```
//dvd.cpp
#include "dvd.h"
#include <iostream>
#include <string>
using namespace std;

DVD::DVD(const string &t, int m, double p) : title(t), minutes(m), price(p) {
}

void DVD::output() const {
    cout << "Title: " << title << '\n' << "Minutes: " << minutes <<
     '\n' << "Price: " << price << '\n';
}

void DVD::changeprice(double p_new) {
    price = p_new;
}
```

```
//A.cpp
#include "dvd.cpp"

int main() {
    const DVD diehard("Die Hard", 130, 9.99);
    DVD stein("Stein's Gate", 120, 10.11);
    DVD bun("Bunraku", 121, 3.56);

    diehard.output();
    stein.output();
    bun.output();

    stein.changeprice(1.11);
    bun.changeprice(0);

    diehard.output();
    stein.output();
    bun.output();

    cout << diehard.get_title() << '\n';
    cout << stein.get_minutes() << '\n';
    cout << bun.get_price() << '\n';
}
```

```
> g++ -Wall -g driver.cpp -o driver
> ./driver
Number of items in the list: 0

Number of items in the list: 1
7
Number of items in the list: 2
2  7
Number of items in the list: 3
5  2  7
8 could not be removed.
2 was successfully removed.
Number of items in the list: 2
5  7
7 was successfully removed.
Number of items in the list: 1
5
5 was successfully removed.
Number of items in the list: 0

8 could not be removed.
Number of items in the list: 0
```

```
//Node.h
class Node{
    int data;
    Node * next;
    public:
        Node(int x):data(x),next(0){}
        Node(int x, Node * y):data(x), next(y){}
        int getData() {return data;}
        Node* getNext() {return next;}
        void setNext(Node* p){next = p;}
    };
```

```
//LinkedList.h
class LinkedList{
    Node *head;
    int numberOfItems;
    public:
        LinkedList():head(0),numberOfItems(0){}
        void insert(int);
        bool remove(int);
        int getNumberOfItems() const {return numberOfItems;}
        Node* getHead() const {return head;}
};
```

```cpp
//LinkedList.cpp
#include "Node.h"
#include "LinkedList.h"
#include <iostream>
using namespace std;

void LinkedList::insert(int value) {
    Node *p = new Node(value, head);
    head = p;
    ++numberOfItems;
}

bool LinkedList::remove(int value) {
    Node *current = head;
    Node *previous;
    while (current != 0 && current->getData() != value) {
        previous = current;
        current = current->getNext();
    }

    if (current==0)
        return false;

    if (current == head)
        head = head->getNext();
    else
        previous->setNext(current->getNext());

    --numberOfItems;
    return true;
}

ostream & operator<<(ostream & os, const LinkedList & l) {
    Node *p = l.getHead();
    while(p!=0){
        os << p->getData() << "  ";
        p = p->getNext();
    }

    return  os;
}
```

```
//driver.cpp
#include "LinkedList.cpp"

int main(void){
    LinkedList num;

    cout << "Number of items in the list: " << num.getNumberOfItems()
     << '\n' << num << '\n';

    num.insert(7);
    cout << "Number of items in the list: " << num.getNumberOfItems()
     << '\n' << num << '\n';

    num.insert(2);
    cout << "Number of items in the list: " << num.getNumberOfItems()
     << '\n' << num << '\n';

    num.insert(5);
    cout << "Number of items in the list: " << num.getNumberOfItems()
     << '\n' << num << '\n';

    if (!num.remove(8))
        cout << "8 could not be removed.\n";

    if (num.remove(2))
        cout << "2 was successfully removed.\n";

    cout <<"Number of items in the list: " << num.getNumberOfItems()
     << "\n" << num << '\n';

    if (num.remove(7))
        cout << "7 was successfully removed.\n";

    cout <<"Number of items in the list: " << num.getNumberOfItems()
     << "\n" << num << '\n';

    if (num.remove(5))
        cout << "5 was successfully removed.\n";

    cout <<"Number of items in the list: " << num.getNumberOfItems()
     << "\n" << num << '\n';

    if (!num.remove(8))
        cout << "8 could not be removed.\n";

    cout <<"Number of items in the list: " << num.getNumberOfItems()
     << "\n" << num << '\n';
}
```