Linked List and Node definition for Linked List related questions.

class IntSLList {
public:
    IntSLList(){
        head = tail = NULL;
    }
    ~IntSLList();
    int isEmpty() const{
        return head == NULL;
    }
    void addToHead(int);
    void addToTail(int);
    int deleteFromHead(); // delete the head and return its info
    int deleteFromTail();  // delete the tail and return its info;
    void deleteNode(int);  // delete node containing the value provided as arg;
    bool isInList(int) const;
    void noDuplicates(); //question 1
    void shrink(); //question 2
    void moveToFront(int key); //3
private:
    IntNode *head, *tail;
};

class IntNode {
public:
    int info;
    IntNode *next;
    IntNode(int el, IntNode *ptr = NULL) {
        info = el;
        next = ptr;
    }
};
Q1. Consider a linked list whose members are integer numbers only. (25 P)
Write a method called noDuplicates(). This method should remove all duplicates from the Linked List. That is, if three items with the key 17 appear in the Linked list, noDups() should remove two of them. Consider following example:
Input LList:head->3->41->3->41->2->7->12->7->1->9->12->1->NULL
Resultant list is
Llist:head->3->41->2->7->12->1->9->NULL
Q2. Linked list contains consecutive repeated int numbers as illustrated in example. Write a method shrink() removes these repeated numbers and leave one copy for each sequence of repeated number. (25)
L: 6->6->6->6->5->5->5->10->10->10->10->5->5->5->3->NULL
Resultant List after shrink() method
L: 6->5->10->5->3->NULL
Q3. Write a function `void moveToFirst(int key)` that searches list L for a node that matches a key value. If key exists in L search causes the node to be moved to the beginning of the list. An unsuccessful search leaves the list unchanged. For example, if L={1,2,3,4}, then `moveToFirst(3,L)` produces L={3,1,2,4}. (10P)
Q4. Write a function called `catStack`, that concatenates the contents of first stack on top of second. 

```c
void catStack(Stack *s1, Stack *s2)
```

You may assume int stack class is implemented `pop`, `push`, `itemAtTop` and `isEmpty` methods are available. (15p)

```
6 18
9 5
8 4
S1 S2
```

```
6 18
9 5
8 4
S2
```

Page 5 of 6
Q5. A useful tool for checking your code is to see if the "<" (open) and ">" (close) symbols are properly balanced. For example the sequence \(< / this is valid sequence/>\) is valid but the sequence \(< / test sequence/> / >\) is not. Write a function that takes argument of string type and returns true if string contains balanced "<" and ">" symbols. (Assume you have only character stack class definition with methods push pop isEmpty itemAtTop ).(25p)

```cpp
class charStack{
private:
    int size;
    int *items;
    int top;
public:
    intStack(int sz=100);
    bool isEmpty()const;
    char pop();
    void push(char);
    char itemAtTop()const;
    void clear();
};
```