Programming Project #3
EGRE246 Spring 2019
Linked List

1 Project Specification

Implement the C++ non-ordered `LList` class as specifed below. You should submit only your `.cpp` implementation file. You may work in pairs for this project.

File `LList.h` (available on the class web pages):

```
#ifndef LLIST_H
#define LLIST_H

namespace egre246 {

  class LList {
  public:
    typedef int value_type;
    typedef size_t size_type;
  private:
    ////////////////////////////
    class LLNode {
    public:
      value_type data;
      LLNode* next;

      LLNode(const value_type& data_ = value_type(),
             LLNode* next_ = nullptr) {
        data = data_; next = next_;
      }
      LLNode(LLNode& node) {
        data = node.data; next = node.next;
      }
    };
    ////////////////////////////

    LLNode *head, *tail, // head points to first node in list, tail the last node;
                         // head = tail = nullptr when list is empty
      *curr; // for iterator
    size_type size;
    void clr(LLNode*); // deallocates all nodes in list

  public:
    LList();
    LList(const LList&); // copy constructor
```

```
    ~LList() { clear(); }

    int getSize() const;
    bool isEmpty() const;

    void clear();               // deallocates all nodes; always invalidates iterator
    value_type get(int) const;  // argument is index of value to return (with 0 as first
                                // item); result of using an out-of-bounds index is
                                // undefined
    void add(const value_type); // adds to end of list; always invalidates iterator
    void remove(const int);     // argument is index of value to remove (with 0 as first
                                // item); invalidates iterator if item is removed; does
                                // nothing if index is not valid
    int find(const value_type) const; // returns index of item found or -1 if item is
                                      // not found

    std::string toString() const; // format: "1,2,3", or returns empty string for empty list
    value_type& operator[](int i); // does not invalidate iterator; result of using an
                                   // out-of-bounds index is undefined; index 0 is first item
    LList& operator=(LList&);      // lhs iterator is set to off list
    bool operator==(const LList&); // order and current position for iterators are irrelevant

    // iterator routines; uses position of curr where relevant
    bool isOnList();
    void begin();
    void advance();   // must be on list to advance
    value_type get(); // result returned is undefined if not on list
    void insertAfter(const value_type);
    void insertBefore(const value_type);
    void remove();

    friend std::ostream& operator<<(std::ostream&, const LList&);
    // format: [1,2,3] , or [] for empty string
};

  std::string toString(const LList::value_type&); // same format as LList::toString

}
#endif
```

2 Deliverables

You are to turn in your project through the project submission link on the class web page. Name your class source code file `proj3`*XXXX*`.cpp` where *XXXX* is the last 4 digits of your student id number. If two of you work on the project only submit one file using one of your id numbers with

both of your names in the comments at the top of the file. You should not turn in any driver programs. You must also document your class as in the previous projects.

Due date: April 2