Lab 10
Implementation and Applications of Linked List

Section 1: Guess program outputs.

For each of the following program fragments, predict what the output will be.

1.

```
struct ListNode
{
  double value;
  ListNode *next;

  ListNode(double value1, ListNode *next1 = nullptr)
  {
    value = value1;
    next = next1;
  }
};

ListNode *p = new ListNode(56.4);
p = new ListNode(34.2, p);
cout << (*p).value << endl << p->value;
```

2.

```
ListNode *p = new ListNode(56.4);
p = new ListNode(34.2, p);
ListNode *q = p->next;
cout << q->value << endl;
```

3.

```
ListNode *p = new ListNode(56.4, new ListNode(31.5));
ListNode *q = p;
while (q->next->next != nullptr)
  q = q->next;
cout << q->value;
```

Section 2: Review Questions and Exercises

1. Using the ListNode structure introduced in the section 1, write a function
void printFirst(ListNode* ptr)
that prints the value stored in the first node of a list passed to it as parameter. The function should print an error message and terminate the program if the list passed to it is empty.

2. Write a function
void printSecond(ListNode* ptr)
that prints the value stored in the second node of a list passed to it as parameter. The function should print an error message and terminate the program if the list passed to it has less than two nodes.

3. Write a recursive function
double lastValue(ListNode* ptr)
that returns the value stored in the last node of a nonempty list passed to it as parameter. The function should print an error message and terminate the program if the list passed to it is empty.

4. Write a function
ListNode* removeFirst(ListNode* ptr)
that is passed a linked list as parameter and returns the tail of the list: That is, it removes the first node and returns what is left. The function should deallocate the storage of the removed node. The function returns nullptr if the list passed to it is empty.

5. Write a function
ListNode* ListConcat(ListNode* list1, ListNode* list2)
that concatenates the items in list2 to the end of list1 and returns the resulting list.

Section 3: Programming Challenges

1. List Member Deletion

Modify the list class you created in the previous lab by adding a function to remove an item from the list:

void remove(double x);

Test the class by adding a sequence of instructions that mixes operations for adding items, removing items, and printing the list.