CSC 212 Tutorial #4 Solution
List & DLL

Problem 1

```java
public void reverse() {
    if (head == null || head.next == null)
        return;
    Node<T> prev = null, cur = head, next = null;
    while (cur != null) {
        next = cur.next;
        cur.next = prev;
        prev = cur;
        cur = next;
    }
    head = prev;
}

public static <T> void reverse(List<T> list) {
    if (list.empty())
        return;
    list.findFirst();
    int count = 1;
    while (!list.last()) {
        count++;
        list.findNext();
    }
    if (count == 1)
        return;
    list.findFirst();
    T left = null, right = null;
    for (int i = 0; i < (count / 2); i++) {
        left = list.retrieve();
        for (int j = i; j < count - i - 1; j++)
            list.findNext();
        right = list.retrieve();
        list.update(left);
        list.findFirst();
        for (int j = 0; j < i; j++)
            list.findNext();
        list.update(right);
        list.findNext();
    }
}
```

Problem 2

```
public static <T> void circularLeftShift(List<T> list, int n) {
    list.findFirst();
    if (list.last())
        return;
    for (int i = 0; i < n; i++) {
        list.findFirst();
        T elem = list.retrieve();
        list.remove();
        while (!list.last())
            list.findNext();
        list.insert(elem);
    }
}
```

Problem 3

```
public void removeBetween(T e1, T e2) {
    Node<T> p = head;
    while ((p != null) && (!p.data.equals(e1)))
        p = p.next;
    if (p == null)
        return;
    Node<T> q = p.next;
    while ((q != null) && (!q.data.equals(e2)))
        q = q.next;
    if (q == null)
        return;
    p.next = q;
    q.previous = p;
    current = head;
}
```

Problem 4

```
public static <T> void reverseCopy(DoubleLinkedList<T> l1,
   DoubleLinkedList<T> l2) {
    if (l1.empty())
        return;
    while (!l1.last())
        l1.findNext();
    while (!l1.first()) {
        l2.insert(l1.retrieve());
        l1.findPrevious();
    }
    l2.insert(l1.retrieve());
}
```