CMSC 206: Data Structures
Iterators, Stacks, Queues, and Priority Queues

1. For each labeled line of code below, draw the contents of the linked list `list` after that line is run. Underline the element whose data is the most recently returned from the `next` method of `itor`, if applicable.

```
   LinkedList<String> list = new LinkedList<String>();
   list.addLast("a");
   list.addLast("b");
   list.addLast("c");
a. list.addLast("d");


   ListIterator<String> itor = list.listIterator();
b. itor.next();


c. itor.remove();


   itor = list.listIterator();
d. itor.next();


e. itor.next();


f. itor.set("e");


g. itor.add("f");


h. itor.next();


i. itor.add("g");
```

2. After each labeled line, write out what the contents of the stack are. List the contents in order from next-to-be-popped to last-to-be-popped:

```
   Stack<Integer> s = new Stack<Integer>();
   s.push(3);
a. s.push(4);


b. s.pop();


   s.push(10);
   s.push(12);
c. s.push(8);


d. s.push(s.pop() + s.pop());
```

3. After each labeled line, write out what the contents of the queue are. List the contents in order from next-to-be-dequeued to last-to-be-dequeued.

```
   Queue<Integer> q = new LinkedList<Integer>();
   q.add(3);
a. q.add(4);


b. q.remove();


   q.add(10);
   q.add(12);
c. q.add(8);


d. q.add(q.remove() + q.remove());
```