CSE 143, Winter 2009
Sample Midterm Exam #1 Key

1. Two solutions are shown.

```
public void compressDuplicates(Stack<Integer> s) {
    Queue<Integer> q = new LinkedList<Integer>();
    while (!s.isEmpty()) {
        q.add(s.pop());          // s -> q
    }
    while (!q.isEmpty()) {
        s.push(q.remove());      // q -> s, to reverse the stack order
    }
    while (!s.isEmpty()) {       // s -> q
        q.add(s.pop());
    }
    if (!q.isEmpty()) {          // q -> s, replacing dupes with (count, val)
        int last = q.remove();
        int count = 1;
        while (!q.isEmpty()) {
            int next = q.remove();
            if (next == last) {
                count++;
            } else {
                s.push(count);
                s.push(last);
                count = 1;
                last = next;
            }
        }
        s.push(count);
        s.push(last);
    }
}
```

```
public void compressDuplicates(Stack<Integer> s) {
    Queue<Integer> q = new LinkedList<Integer>();
    s2q(s, q);
    q2s(q, s);
    s2q(s, q);
    if (!q.isEmpty()) {          // q -> s, replacing dupes with (count, val)
        int last = q.remove();
        int count = 1;
        while (!q.isEmpty()) {
            int next = q.remove();
            if (next == last) {
                count++;
            } else {
                s.push(count);
                s.push(last);
                count = 1;
                last = next;
            }
        }
        s.push(count);
        s.push(last);
    }
}
```

2.

```
public static int countInAreaCode(Map<String, String> numbers, String areaCode) {
    Set<String> uniqueNumbers = new HashSet<String>();
    for (String name : numbers.keySet()) {
        String phoneNumber = numbers.get(name);
        if (phoneNumber.startsWith(areaCode)) {
            uniqueNumbers.add(phoneNumber);
        }
    }
    return uniqueNumbers.size();
}
```

3.

```
list2.next.next.next = list;     // 4 -> 1
list.next = list2;               // 1 -> 2
list = list2.next.next;          // list -> 4
list2 = list2.next;              // list2 -> 3
list2.next = null;               // 3 /
list.next.next.next = null;      // 2 /
```

4.

```
public boolean isSortedBy(int n) {
    if (n <= 0) {
        throw new IllegalArgumentException();
    }
    ListNode current1 = front;
    ListNode current2 = front;
    while (current2 != null && n > 0) {
        current2 = current2.next;
        n--;
    }
    while (current2 != null) {
        if (current1.data > current2.data) {
            return false;
        }
        current1 = current1.next;
        current2 = current2.next;
    }
    return true;
}
```

5.

Call	Value Returned
mystery(7)	8
mystery(42)	53
mystery(385)	496
mystery(-790)	-801
mystery(89294)	90305

6.

```
public int digitMatch(int x, int y) {
    if (x < 0 || y < 0) {
        throw new IllegalArgumentException();
    } else if (x < 10 || y < 10) {
        if (x % 10 == y % 10) {
            return 1;
        } else {
            return 0;
        }
    } else if (x % 10 == y % 10) {
        return 1 + digitMatch(x / 10, y / 10);
    } else {
        return digitMatch(x / 10, y / 10);
    }
}
```