CSE 143 Section Handout #13
Practice Midterm #5
Solutions

1.

List	Output
(a) `[10, 20, 30]`	`[31, 21, 11]`
(b) `[8, 2, 9, 7, 4]`	`[5, 8, 10, 3, 9]`
(c) `[-1, 3, 28, 17, 9, 33]`	`[34, 10, 18, 29, 4, 0]`

2. Two solutions are shown.

```
public static void filterRange(ArrayList<Integer> list, int min, int max) {
    for (int i = 0; i < list.size(); i++) {
        if (list.get(i) >= min && list.get(i) <= max) {
            list.remove(i);
            i--;
        }
    }
}

public static void filterRange(ArrayList<Integer> list, int min, int max) {
    for (int i = list.size() - 1; i >= 0; i--) {
        int element = list.get(i);
        if (min <= element && element <= max) {
            list.remove(i);
        }
    }
}
```

3.

```
public static int removeMin(Stack<Integer> s) {
    Queue<Integer> q = new LinkedList<Integer>();
    int min = s.pop();
    q.add(min);
    while (!s.isEmpty()) {          // s -> q, looking for min value
        int next = s.pop();
        if (next < min) {
            min = next;
        }
        q.add(next);
    }
    while (!q.isEmpty()) {          // q -> s, filtering out occurrences of min
        int next = q.remove();
        if (next != min) {
            s.push(next);
        }
    }
    s2q(s, q);                      // s -> q -> s, to un-reverse the order
    q2s(q, s);
    return min;
}
```

4.

```
public static Set<String> whoPassed(Map<String, Integer> students,
        Map<Integer, Double> grades, double desired) {

    Set<String> result = new TreeSet<String>();
    for (String name : students.keySet()) {
        int grade = students.get(name);
        if (grades.get(grade) >= desired) {
            result.add(name);
        }
    }
    return result;
}
```

CSE 143 Section Handout #13
Solutions (continued)

5.
```
ListNode list2 = list.next.next.next;   // list2 -> 4
list2.next = list;                      // 4 -> 1
list.next.next.next = list.next;        // 3 -> 2
list = list.next.next;                  // list -> 3
list.next.next = null;                  // 2 /
list2.next.next = null;                 // 1 /
```

6.
```
public boolean hasAlternatingParity() {
    if (front != null) {
        ListNode current = front;
        while (current.next != null) {
            if (current.data % 2 == current.next.data % 2) {
                return false;
            }
            current = current.next;
        }
    }
    return true;
}
```

7.
```
public class Pokemon implements Comparable<Pokemon> {
    ...
    public int compareTo(Pokemon other) {
        if (aggressive) {
            if (!other.isAggressive()) {
                return 1;
            } else {  // both aggressive; break ties by attack+defense power
                return attack + defense
                     - other.getAttack() - other.getDefense();
            }
        } else {  // I am defensive
            if (other.isAggressive()) {
                return -1;
            } else {  // both aggressive; break ties by defense power
                return defense - other.getDefense();
            }
        }
    }
}
```

8.

(a) Indexes examined: 7, 3, 1, 0
Value returned: -1

(b) Initial array: {45, 78, 89, 34, 23, 12, 67, 56}
after 1 pass: {**12**, 78, 89, 34, 23, **45**, 67, 56}
after 2 passes: {12, **23**, 89, 34, **78**, 45, 67, 56}
after 3 passes: {12, 23, **34**, **89**, 78, 45, 67, 56}

(c)
```
{45, 78,  89, 34,  23, 12,  67, 56}
{45, 78,  89, 34} {23, 12,  67, 56}
{45, 78} {89, 34} {23, 12} {67, 56}
{45}{78} {89}{34} {23}{12} {67}{56}
{45, 78} {34, 89} {12, 23} {56, 67}
{34, 45, 78, 89}  {12, 23, 56, 67}
{12, 23, 34, 45, 56, 67, 78, 89}
```

CSE 143 Section Handout #13
Solutions (continued)

9.

Call	Output
`mystery(13);`	`13`
`mystery(42);`	`42, 21`
`mystery(40);`	`40, 20, 10, 5`
`mystery(60);`	`60, 30, 15`
`mystery(48);`	`48, 24, 12, 6, 3`

10.

```
public static int indexOf(String source, String target) {
    if (target.length() > source.length()) {
        return -1;
    } else if (source.substring(0, target.length()).equals(target)) {
        return 0;
    } else {
        int pos = indexOf(source.substring(1), target);
        if (pos == -1) {
            return pos;
        } else {
            return pos + 1;
        }
    }
}
```