CSE142 Section Handout #9 Solutions

1. Reference Mystery. The program produces the following output.

```
108 [5, 10, 14]
8 [5, 10, 14]
20 8
```

2.

Original List	Final List
{4, 1, 3}	{4, 1, 2}
{2, 1, 3, 2}	{2, 1, 2, 4}
{3, 6, 2, 9, 5}	{3, 6, 2, 4, 6}
{1, 1, 1, 1, 8}	{1, 1, 2, 1, 6}
{1, 3, 4, 6, 2, 9}	{1, 3, 2, 4, 2, 8}

3. Inheritance. The output produced is as follows.

```
a          c          a          c
a 1        c 1        a 1        d 1
b 2        c 2        c 2        c 2
```

4. Token-Based File Processing. One possible solution appears below.

```
public static void reportScore(Scanner input) {
    int score = 0;
    int count = 0;
    while (input.hasNextInt()) {
        int next = input.nextInt();
        String type = input.next();
        count += next;
        if (type.equals("*")) {
            score = score + next;
        } else {
            score = score - next;
        }
    }
    String name = input.next();
    System.out.println(name + " got " + score + " of " + count);
}
```

5. Line-Based File Processing. One possible solution appears below.

```
public static void reverseAndFlip(Scanner input) {
    while (input.hasNextLine()) {
        String first = input.nextLine();
        if (input.hasNextLine()) {
            String second = input.nextLine();
            for (int i = second.length() - 1; i >= 0; i--) {
                System.out.print(second.charAt(i));
            }
            System.out.println();
        }
        System.out.println(first);
    }
}
```

6. Arrays. One possible solution appears below.

```
public static boolean isPairwiseSorted(int[] list) {
    for (int i = 0; i < list.length - 1; i += 2) {
        if (list[i] > list[i + 1]) {
            return false;
        }
    }
    return true;
}
```

7. ArrayLists. Two possible solutions appear below.

```
public static void reverse3(ArrayList<Integer> list) {
    for (int i = 0; i < list.size() - 2; i += 3) {
        int n1 = list.get(i);
        int n3 = list.get(i + 2);
        list.set(i, n3);
        list.set(i + 2, n1);
    }
}

public static void reverse3(ArrayList<Integer> list) {
    for (int i = 0; i < list.size() - 2; i += 3) {
        list.add(i, list.remove(i + 2));
        list.add(i + 2, list.remove(i + 1));
    }
}
```

8. Critters. One possible solution appears below.

```
public class Ferret extends Critter {
    private int infectCount;
    private Random r;

    public Ferret() {
        r = new Random();
    }

    public Action getMove(CritterInfo info) {
        if (infectCount > 0) {
            infectCount--;
        }
        if (info.getFront() == Neighbor.OTHER) {
            infectCount = 5;
            return Action.INFECT;
        } else if (info.getFront() == Neighbor.EMPTY) {
            return Action.HOP;
        } else {
            int choice = r.nextInt(2);
            if (choice == 0) {
                return Action.LEFT;
            } else {
                return Action.RIGHT;
            }
        }
    }

    public Color getColor() {
        if (infectCount > 0) {
            return Color.RED;
        } else {
            return Color.BLUE;
        }
    }

    public String toString() {
        return "I=" + infectCount;
    }
}
```

9. Arrays. One possible solution appears below.

```
public static int[] splice(int[] list, int from, int to) {
    int[] result = new int[list.length];
    int index = 0;
    for (int i = to; i < list.length; i++) {
        result[index] = list[i];
        index++;
    }
    for (int i = from; i < to; i++) {
        result[index] = list[i];
        index++;
    }
    for (int i = 0; i < from; i++) {
        result[index] = list[i];
        index++;
    }
    return result;
}
```

10. Programming. One possible solution appears below.

```
public static boolean isMatch(String pattern, String text) {
    int j = 0;
    for (int i = 0; i < pattern.length(); i++) {
        char ch1 = pattern.charAt(i);
        if (ch1 == '*') {
            int diff = text.length() - pattern.length() + 1;
            if (diff < 0) {
                return false;
            } else {
                j += diff;
            }
        } else {
            if (j >= text.length()) {
                return false;
            }
            char ch2 = text.charAt(j);
            j++;
            if (ch1 != '.' && ch1 != ch2) {
                return false;
            }
        }
    }
    return j == text.length();
}
```

below is a solution to the 6-point problem:

```
public static boolean isMatch(String pattern, String text) {
    if (text.length() != pattern.length()) {
        return false;
    }
    for (int i = 0; i < pattern.length(); i++) {
        char ch1 = pattern.charAt(i);
        char ch2 = text.charAt(i);
        if (ch1 != '.' && ch1 != ch2) {
            return false;
        }
    }
    return true;
}
```