ArrayList (5 points)

```
[31, 21, 11]
[5, 8, 10, 3, 9]
[34, 10, 18, 29, 4, 0]
```

Map Mystery (5 points)

```
[ok, dog, horse, horse]
[fruit, hyena, bird, hello, hello]
[hhh, gg, e]
```

Recursive Tracing (5 points)

```
13
42, 21
40, 20, 10, 5
60, 30, 15
48, 24, 12, 6, 3
```

Inheritance Mystery (5 points)

`var1.one();`	`Blue 1/Green 1`
`var1.two();`	`compiler error`
`var1.three();`	`Green 3`
`var2.one();`	`Blue 1/Green 1/Red 1`
`var2.two();`	`compiler error`
`var2.three();`	`Red 2/Yellow 2/Yellow 3`
`var3.one();`	`Blue 1/Green 1`
`var3.two();`	`compiler error`
`var3.three();`	`Yellow 2/Yellow 3`
`var4.one();`	`compiler error`
`((Blue)var1).one();`	`Blue 1/Green 1`
`((Yellow)var1).two();`	`runtime error`
`((Red)var2).three();`	`Red 2/Yellow 2/Yellow 3`
`((Object)var3).one();`	`compiler error`
`((Yellow)var3).two();`	`Yellow 2`
`((Green)var4).three();`	`Green 3`
`((Red)var4).one();`	`runtime error`

Programming 1

Grading rubric

Item	Points
Has correct method header	3
Correctly loops through half of `elementData`	3
Addresses odd length case	2
Correctly adjusts `size`	2
Total	10

Sample solutions

```
public void collapse() {
    for (int i = 0; i < size/2; i++) {
        elementData[i] = elementData[2*i] + elementData[2*i + 1];
    }
    if (size%2 == 1) elementData[size/2] = elementData[size - 1];
    size -= size/2;
}
```

```
public void collapse() {
    for (int i = 0; i < size; i += 2) {
        elementData[i - i/2] = elementData[i] + elementData[i + 1];
    }
    size -= size/2;
}
```

Programming 2

Grading rubric

Item	Points
Has correct method header	3
Has different length base case	2
Has length less than 1 base case	2
Has a recursive call with substrings	3
Total	10

Sample solutions

```
public static boolean isReverse(String s1, String s2) {
    if (s1.length() != s2.length()) return false;
    if (s1.length() <= 1) return true;
    if (s1.toLowerCase().charAt(0) == s2.toLowerCase().charAt(s2.length() - 1))
        return isReverse(s1.substring(1), s2.substring(0, s2.length() - 1));
    return false;
}

public static boolean isReverseC(String s1, String s2) {
    if (s1.length() == 0 && s2.length() == 0) {
        return true;
    if (s1.length() == 0 || s2.length() == 0) {
        return false;  // not same length
    String s1first = s1.substring(0, 1);
    String s2last = s2.substring(s2.length() - 1);
    return s1first.equalsIgnoreCase(s2last) &&
        isReverse(s1.substring(1), s2.substring(0, s2.length() - 1));
}
```