Processing arrays recursively

Add up all the elements in an array:

Iterative version:

```
int sum = 0;
for (int i = 0; i < array.size(); i++) {
    sum += array[i];
}
```

Recursive version (pseudocode):

Base case: If array.size() == 1, return array[0].
Recursive case: If array.size() > 1:

- Recursively compute the sum of all the elements in the sub-array from index 1 to the end.
  in Python this would be `sum(array[1:])`
- Add array[0] to the sum from the previous step.
- Return this value.

Code:

```
public static int sumOfArrayList(ArrayList<Integer> list)
{
    return sumOfArrayList(list, 0);
}

// Helper function for above.  Assume leftIdx is the left most index of the "sub-list"
// we are computing the sum of.  (So we are summing elements from leftIdx through the end.)

private static int sumOfArrayList(ArrayList<Integer> list, int leftIdx)
{
    if (leftIdx == list.size()-1) {  // if there's only one element in our "sub-list"
        return list.get(leftIdx);
    }
    else {
        int smallerSum = sumOfArrayList(list, leftIdx + 1);
        return list.get(leftIdx) + smallerSum;
    }
}
```

Find the maximum element in an array:

Iterative pseudocode:

```
largest = array[0]
for (int i = 1; i < array.size(); i++)
   if (array[i] > largest)
      largest = array[i]
return largest
```

Recursive pseudocode:

Base Case:

Recursive Case: