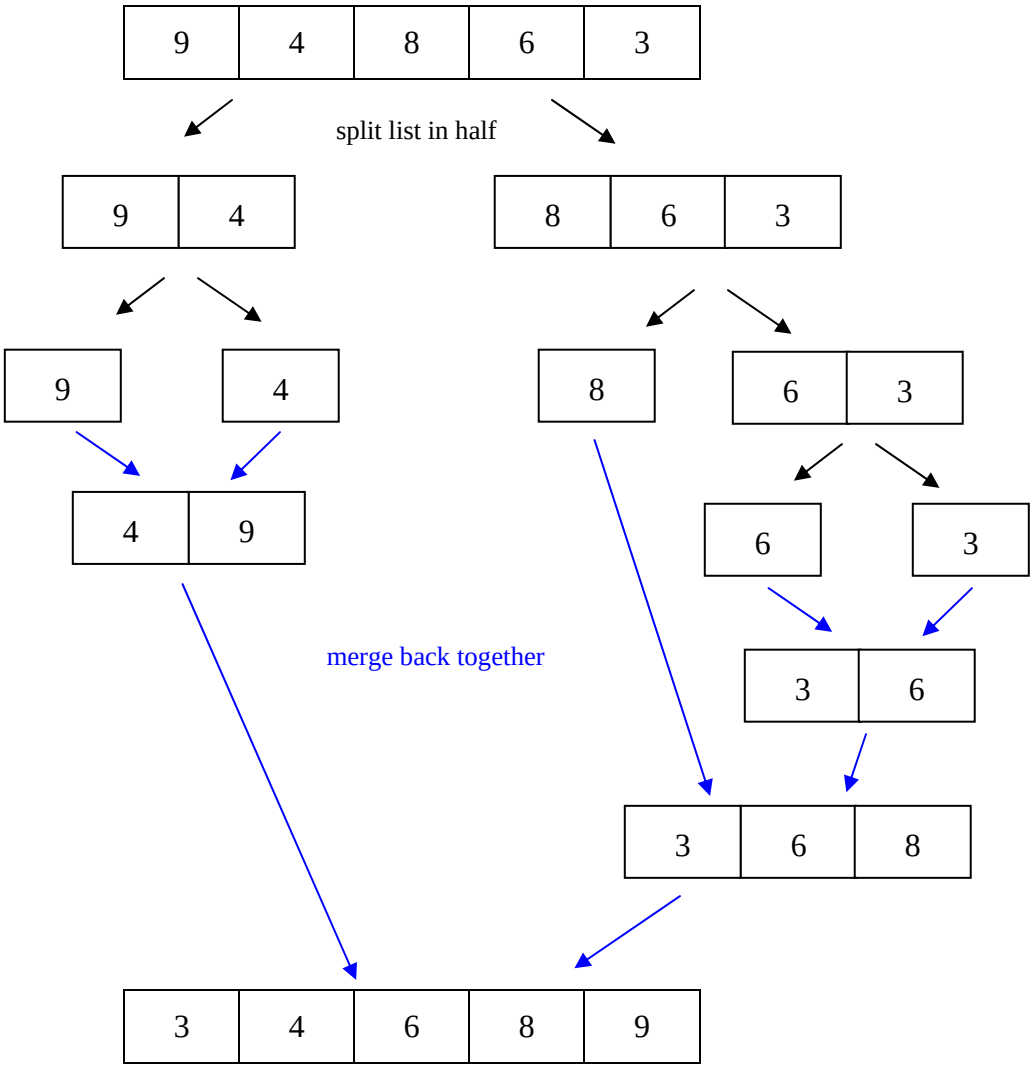
Merge Sort (on AP Test)

1. If the size of the array is not larger than 1
   a. Assume an array of size 1 or 0 is in order.
2. If the size of the array is larger than 1
   a. Split the list in half.
   b. recursively Merge Sort each half of the list.
   c. Merge the sorted halves back together in order.

Merge Sort Example



Merge Sort Algorithm

```
public static int[] mergeSort(int[] array)
{
     if(array.length<=1)
          return array;
     else
     {
          //copy first half
          int[] arr1 = new int[array.length/2];
          System.arraycopy(array, 0, arr1, 0, arr1.length);

          //copy second half
          int arr2length = array.length - arr1.length;
          int[] arr2 = new int[arr2length];
          System.arraycopy(array, arr1.length, arr2, 0, arr2length);

          //recursively sort each half
          arr1 = mergeSort(arr1);
          arr2 = mergeSort(arr2);

          //merge sorted arrays
          merge(arr1, arr2, array);
          return array;
     }
}

public static void merge(int[] arr1, int[] arr2, int[] array)
{
     int arr1Counter = 0;
     int arr2Counter = 0;
     int sortedCounter = 0;

     //while neither sub-array is done, fill the sorted one
     while(arr1Counter<arr1.length && arr2Counter<arr2.length)
     {
          if(arr1[arr1Counter]<arr2[arr2Counter])
               array[sortedCounter++] = arr1[arr1Counter++];
          else
               array[sortedCounter++] = arr2[arr2Counter++];
     }
     //finish filling sorted array with whatever is left and return
     while(arr1Counter<arr1.length)
          array[sortedCounter++] = arr1[arr1Counter++];

     while(arr2Counter<arr2.length)
          array[sortedCounter++] = arr2[arr2Counter++];
}
```

Merge Sort Problems

Because the new sub-arrays at each recursive call to mergeSort all have to be stored in memory, Merge Sort can end up being a very costly algorithm.

You can re-write the algorithm more efficiently to stop the excessive array copying (i.e., all the calls to the System class's arraycopy method), but you can't easily get rid of the additional memory requirements caused by the algorithm's recursive nature.