**Question 1. Sorting**

In this question you will implement two popular sorting algorithms: HeapSort and MergeSort. You can run them on the same Integer input and compare their running times on your computer to Arrays.sort, which uses an optimized QuickSort (see the test program testYorkArrays).

You will implement your sorting algorithms as static methods provided by the the new utility class YorkArrays.

Your implementation of HeapSort should be very concise, since you will make use of the heap-based PriorityQueue implementation provided by java.util. While this will make your code short, note that you will pay the price that your implementation will not be in-place, since PriorityQueue allocates its own heap – it does not use the input array for the heap.

In total, you will implement three methods for sorting an array in non-decreasing order:

> **heapSort (a)** – Sorts the Integer array a, returning the result in a.
> **mergeSort (a, p, q)** - Sorts the Integer subarray a[p…q], returning the result.
> **merge (a, b)** – Merges the two input Integer arrays a and b and returns the result.

Remember to test your program on a broad range of test cases. Pay particular attention to boundary conditions.