ETEC2101	Lab7: **Heaps**	Points: **100**
Assigned: **11/21/2016**	Due: **12/2/2016 (Friday) @ 3pm**	***(115 possible)***

Tasks:

- (**30 points**) Make a Generic **BinaryHeap** (the generic will need to extend Comparable) as discussed in class
  - Make sure the user is able to specify (somehow) that they want a "min-heap" or a "max-heap"
  - At a minimum, define a **push**, **pop**, and **size** method.
  - Two constructors: one that creates an empty heap, one that takes a Java *Collection* (look it up) of pre-existing values.
  - (**10 points**) Style, docstrings, good variable names, etc.
  - (**10 points**) Make some kind of simple test program that thoroughly tests all your BinaryHeap functionality.
- (**20 points**) Make a **WordCounter** class that:
  - reads all words from an arbitrary text file (I'd make it a large one). It should only consider alphabetic characters (A – Z and a – z) and should not consider case.
  - I would suggest storing these in a HashMap (Strings => Integers) where the Integer is the number of times that word occurs in the file.
  - (**15 points**) *Without* using the BinaryHeap, find the top n words in the file. Calculate and output the time taken in seconds (I used System.currentTimeMillis for this).
  - (**15 points**) Now use the BinaryHeap to do the same thing (you'll like need to define a "Pair" class [Word + Count]). Also output the time – this should be significantly less than the "brute-force" version.
- (**15 points**) Make your BinaryHeap class capable of modifying a array of Comparables (I'd suggest giving some static functions for this purpose) and that *doesn't* copy them to your internal format.