Lab #2: C++ Introduction Lab
Due: April 21, 2010 11:59pm

Overview
For this lab you will implement a binary min heap. A heap uses a tree-like structure to maintain an ordering on its elements. A min heap provides for quick access to the smallest element in a collection of values. In addition, the structure of the heap allows for relatively quick insertion and extraction of values while maintaining the heap properties.

A binary min heap can be used as a substitute for a priority list. Such a data structure can be used for ordering events based on the time to trigger.

Algorithmic details can be found in a data structures or algorithms book or on Wikipedia (the article for binary heaps, specifically, is well-written).

You cannot reference any actual implementations (the purpose of this lab is not to translate from another language or to see if another C++ implementation looks like what you were thinking of doing).

Details

• Create a BinaryMinHeap class.

• Your binary min heap will store only integer values (you may choose to templatize your implementation if you wish, but you are not required to do so).

• Your binary min heap will be implemented using an array but there are no restrictions on the number of elements that can be inserted. As such, the backing array will need to grow as needed.

• Your binary min heap must support for the following operations.
  void insert(int) inserts the provided integer into the binary min heap.
  int getMin() returns the smallest value stored in the binary min heap but does not remove the value.
  void removeMin() removes the smallest value from the binary min heap.

• If either getMin or removeMin is invoked on an empty heap, raise an EmptyHeap exception (you should define this exception class).