Lab 4 Heap

Assignment Overview

Write a program that implements heap class and test the heap inserts and deletes.

Before doing the assignment, be sure to read Course Guidelines document in D2L Content.

Heap Class

See *Heap-pseudocode.txt* for a partial pseudocode of **Heap** class.

Heap class shall have at least the following methods

- heap() default constructor
- insert(element) that inserts the element via percolate up
- deleteMin() that deletes the minimum element via percolate down
- getElement(index) that returns the heap vector element at the index
- size() that returns the heap size

If you are comfortable with it, implement a generic (template) class. In such a case, as **Comparable** type, use **int,** if the language allows it, or **Integer** or similar wrapper type, otherwise.

Driver (main)

Write main driver class/function that uses the heap class, tests it, and prints test results. See *Driver-pseudocode.txt* file for the driver pseudocode.

The driver has to process all provided input files. For each input file, the driver has to create a heap and to

- read the input file data into a dynamic array (vector)
- insert the data into the heap, in the same order as in the input file
- insert 31 into the heap
- insert 14 into the heap
- delete all data from the heap via deleteMin()

Look in testData() function in the pseudocode to see how the insertion/deletion shall be implemented.

In each test, after inserting/deleting of every element, the driver has to verify that the heap object is really a heap by checking that each element is less than or equal to its children. See checkHeap() function in the pseudocode.

When printing the heap elements, print the heap size first – see printHeap() function. After the size, print all elements of the underlying heap vector.

Input and Output Files

Each input file contains heap elements separated by whitespaces. All keys have to be read and processed until the end of file.

	Correspondent files	**Correspondent files**	**Correspondent files**	**Correspondent files**
Provided test input files	in1.txt	in2.txt	in3.txt	in4.txt
Provided sample output files	out1_sample.txt	out2_sample.txt	out3_sample.txt	**none**
Output files to produce	*out1.txt*	*out2.txt*	*out3.txt*	*out4.txt*

Use the provided files to test your program. Run the program using all four test input files to produce four output files and submit the input and output files together with the source code.

The format of the produced output files shall be similar to the provided sample output files. The heaps have to be the same.