Homework 4 - Heaps

Write a program (using the textbook code provided in my git repository) to take *N* elements, of whatever kinds of objects you'd like) and do the following:

a. Insert them into a heap one by one.
b. Build a heap in linear time using buildHeap.

(You'll get bonus points if you use something more interesting than Integers for this)

Compare and provide an analysis of the running time of both algorithms for sorted, reverse-ordered, and randomly-ordered inputs. You'll obviously have to run this a bunch of times with at least 8-10 different values of N. Try ranges between one thousand and five million. How does the timing depend on N, and the ordering of the data at the start? Plot the data in Excel or a similar program to do your analysis, with N as the x axis and time as the y axis. Discuss and analyze your results and compare with your expectation based on the algorithm analysis in your textbook. Upload this analysis as a PDF into your git repository, just as you would with a source-code file.

NOTE: You'll want your computer to be as "un-busy" as possible while doing your runs, so that the timing is not affected too much by other running code attempting to share your processor. Your best bet is to attempt your runs after a reboot with everything else but your bash shell closed….

Below is an example of how to do timing of an algorithm in a java program:

```
import java.util.concurrent.TimeUnit;


class TimeExample

{

        public static void main(String[] args) throws InterruptedException {


                long start = System.nanoTime();


                // ... the code being timed goes here ...


                // as an example: sleep for 5 seconds

                TimeUnit.SECONDS.sleep(5);


                // ... the code being timed ends ...


                long end = System.nanoTime();
```

```
        long msElapsed = (end - start) / 1000000;


        System.out.println("Execution time in milliseconds: " + msElapsed);
    }
}
```