Name: Ibrahim Kamal
Course: SE 320
Homework 4 - Performance Evaluation

1. **A short (one paragraph) explanation of why the benchmark harness is structured as it is. Why is there an initial run that isn't counted in the min/max/average? Why are we measuring multiple runs?**

   The initial run is used for warming up the cache, if we don't do so there would be one run which will be much slower than the others. Since this run will be much slower than the others it is not beneficial to use this value while calculating min/max/avg time. Min/max/avg times should be calculated once the system is warmed up and some similar data/value has been loaded into the caches.

   Multiple runs are required because we might get lucky or unlucky in the first run. Min, max, avg times also tells us about how a system will perform in peak times and when the system has minimum load. Apart from this, the inputs to the Red-Black BST is essential in determining the time taken for insert/delete/lookup operation, so it becomes all the more necessary to do multiple runs in order to test the performance of this particular system.

2. **Reasonable operational profiles for use of the RedBlackBST in the following use cases. Each of these profiles, for our purposes, is simply a percentage of insert/delete/lookup operations (3 numbers that sum to 100 in each case).**
   a. **A logging data structure, which mostly records new data, with occasional queries and edits (deletions)**
      insert: 60
      delete: 20
      lookup: 20

   b. **A read-heavy database, with occasional updates and deletions.**
      insert: 20
      delete: 20
      lookup: 60

   c. **A read-only database, with no modifications performed.**
      insert: 0
      delete: 0
      lookup: 100

3. **A description of what steps you took in preparation for your performance measurements. For example, what other software is or isn't running on your machine? (For this assignment, you may ignore our discussion of frequency scaling's effect on performance tests, since disabling it is quite inconvenient and many of you are probably using laptops.)**

   I terminated all the other applications apart from Google Chrome which I was using to write down my answers (on Google docs). I ran the Benchmark.java file on my Macbook Pro with 8 gb of ram.

4. **For each of your operational profiles, report the following, which should be produced by Benchmark.main() once you update the hard-coded operational profile.**

   a. **A logging data structure, which mostly records new data, with occasional queries and edits (deletions)**
      Average for 1000000 operations: 179896813ns
      Minimum for 1000000 operations: 144602500ns
      Maximum for 1000000 operations: 301886519ns

   b. **A read-heavy database, with occasional updates and deletions.**
      Average for 1000000 operations: 168944453ns
      Minimum for 1000000 operations: 102995390ns
      Maximum for 1000000 operations: 296912675ns

   c. **A read-only database, with no modifications performed.**
      Average for 1000000 operations: 16355899ns
      Minimum for 1000000 operations: 13272599ns
      Maximum for 1000000 operations: 23874006ns

5. **Which of your operational profiles has higher throughput (i.e., performs work per unit time)? What about the red-black tree might explain this outcome?**

   The read-only database has the highest throughput which can be explained due to the fact that in a Red-Black BST lookup is faster than insert and delete. Insert and delete takes more time due to the fact that after every insert and delete operation, some rebalancing may be required which is essential to make the BST balanced. Due to the reason stated above logging and read-heavy database take more time than the read-only database and thus have a lower throughput.

6. **Are your warmup times noticeably different from the "main" iteration times? Most likely yes, but either way, why might we expect a significant difference?**

   It is likely because the caches/IO are not properly optimized. The warm-up runs are to make sure that there is some similar type of data/information loaded into caches which would be helpful in the main runs using which we use to calculate the min, max, or avg time.

7. **The numbers for operational profile (c) aren't actually that useful. Look at the benchmarking code, and explain why those numbers, as reported, tell us nothing about the performance of using the BST in a real application for a read-only workload.**

   It is not useful to test a database under read-only conditions as by doing this we can't find any performance issues while writing or deleting data. I think that in a real world, it is not that useful to have only a read-only database, since the data has to be written first in order to read or delete it.

8. **Assume each run (each call to runOps) simulates the activity of one remote request. Based on the average execution time for each operational profile, what would the throughput be for each of your profiles? (i.e., requests/second)**

   a. **A logging data structure, which mostly records new data, with occasional queries and edits (deletions)**
      Average for 1000000 operations: 179896813ns
      Throughput = 1 / **(**179896813 * 10^(-9)) = 5.56 requests/second

   b. **A read-heavy database, with occasional updates and deletions.**
      Average for 1000000 operations: 168944453ns
      Throughput = 1 / **(**168944453 * 10^(-9)) = 5.92 requests/second

   c. **A read-only database, with no modifications performed.**
      Average for 1000000 operations: 16355899ns
      Throughput = 1 / **(**16355899 * 10^(-9)) = 61.1 requests/second

9. **Assuming each remote user makes 5 requests/minute, your program's resource usage scales linearly, and we are only interested in CPU execution time, how many concurrent remote users could you support on your machine (again, do this for each operational profile) without degrading performance or overloading the system?**

   a. **A logging data structure, which mostly records new data, with occasional queries and edits (deletions)**
      Throughput = 1 / **(**179896813 * 10^(-9)) = 5.56 requests/second
      Number of concurrent users = 5.56 * 60 / 5 = 66

   b. **A read-heavy database, with occasional updates and deletions.**
      Throughput = 1 / **(**168944453 * 10^(-9)) = 5.92 requests/second
      Number of concurrent users = 5.92 * 60 / 5 = 71

   c. **A read-only database, with no modifications performed.**
      Throughput = 1 / **(**16355899 * 10^(-9)) = 61.1 requests/second
      Number of concurrent users = 61.1 * 60 / 5 = 733

10. **What aspects of load are we not testing, which could possibly reduce the capacity of your machine to service requests?**

    We are not performing stress testing for any of the applications. We don't know how the application will perform when "stressed" with activity at or beyond peak load conditions.

    We are not performing longevity testing for any of the applications. We don't know how the application will perform when used for a long period of time, under normal and peak load.

    We are not testing if there are any memory leaks or not.