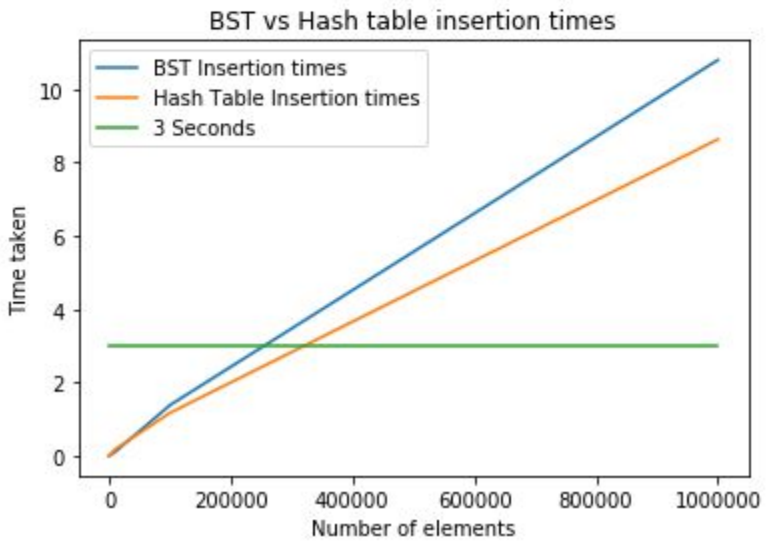
Question 3: Binary Search Tree vs Hash Table insertion times

- *Hypothesis*:
  In the C++ standard library, std::multiset can be used to easily store values in a balanced binary tree. Likewise, std::unordered_multiset can be used to store values in a hash table. In theory, a hash table should be faster for insertions and deletions. We predict this to be true, and for hash table insertions to be roughly 50% faster in large values of n.
- *Methods*:
  We inserted n values into both a BST and a hash table at varying values of n, starting at 10. We then timed how long it took to run n amount of inserts, with n being multiplied by 10 every time.
- *Results*:
  We ran various experiments with different source code, and ended up concluding that at n = 1000000, the hash table was 25% faster (10.799 seconds to insert 1000000 items compared to 8.635 seconds).



- *Discussion*:
  This was about what we had predicted. Something that was surprising was that at n=1,000 and n=10,000, the insertion time for the hash table is slightly slower than the BST. This is likely due to n being too small to get an accurate measurement.
- *Conclusions*:
  Under the conditions tested, insertion time for hash tables is consistently faster than insertion time for binary search trees.