Project #5

Daniel Williamson

12/9/16

Hash Set:

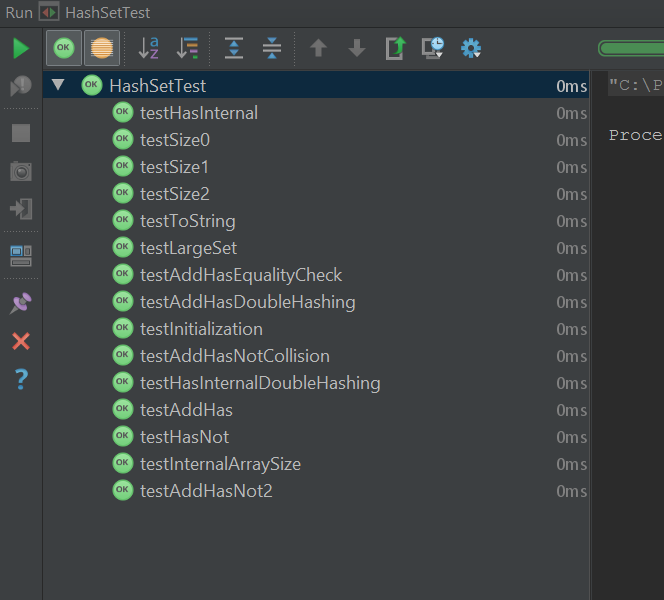
Approach:

My approach for this project was just like many others. I start by doing a lot of research. I reference the book, the internet and peers. Stephen White was one in specific that was a giant help. After figuring out how hashing works, it was a rather simple concept. In hash set’s we use double hashing. Which essentially takes an element, hashes it, and then sees were that value would lie in our array. If for any reason there is already a value at that index, we hash it again. But before we do that, we convert the element into a string, then do our second hash. Which then would give us the position that it belongs.

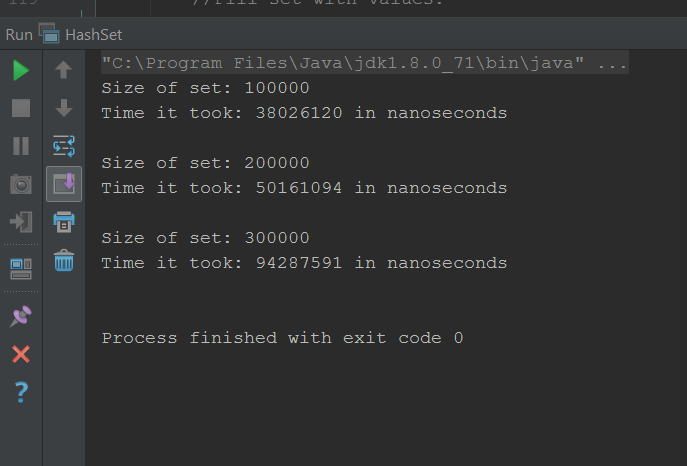
To begin, I needed to understand what the tests were studying for. Basically, we have an add method. Which does exactly as it sounds, it adds the elements to the array. While going to the hashed index it finds that it is not null, which means there is already an element in that position, we then convert it into a string and hash it again. Then insert at that new location. The next method was, has. Has take’s in an element and checks to see if we have that element located somewhere in our array. At first my code wasn’t very efficient. After writing the profilers I found out, it was taking almost a half-hour to check to see if an array has an element. After consulting with my peers, we made a more efficient way of checking. We start with a while loop, if at the index it is null. That means we don’t have it. Simple. But if there is a value there and it is equal to the value we are looking for, we return true. If at any time while looking for an element and we exceed the amount of indexes we have in the array, we know we don’t have that value. Another method we had to complete was size. Size was just a simple for loop that ran the length of the array and returned an integer. Lastly, we had the toString. Which just like previous projects, just goes through all the elements and prints them out in a nice manner.

Here are my results:

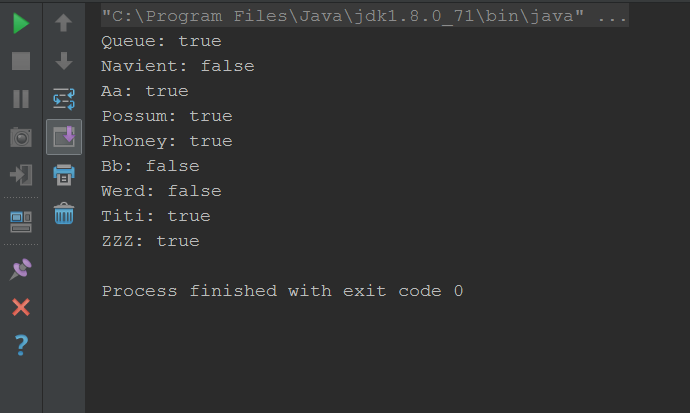
All the tests passing:



The profiler produced these results:



And lastly, the scrabble words:



Hash Map:

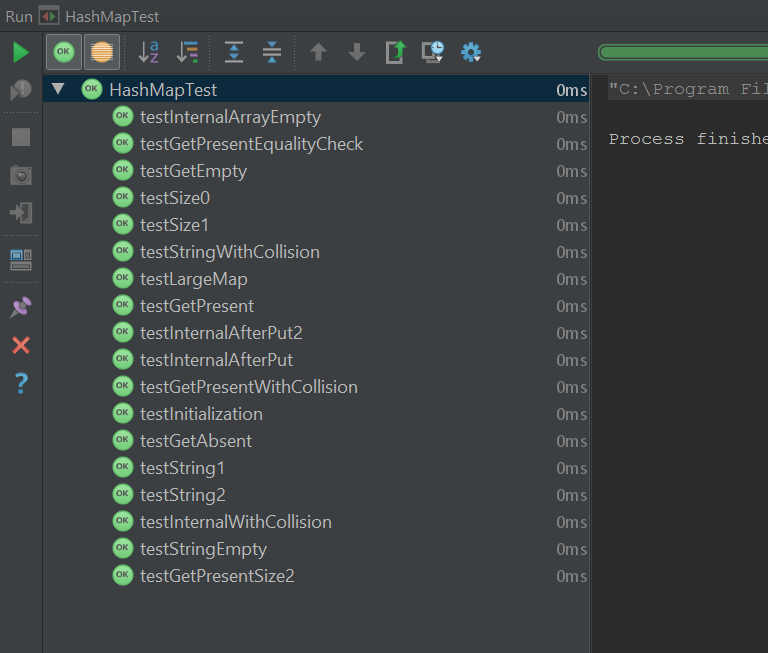
Approach:

Just like I said earlier, I started Hash Map’s with doing some research on how it needs to work. I used the internet, the book and peers. Namely Stephen White. The idea of how Hash Map’s work is rather simple. It is just like a dictionary in the sense that, you have a key and an element. And in Hash Map’s we implemented the separate chaining technic. Which is kind of like a 2D array. It’s an array, that at any index, can hold multiple nodes linked together. Essentially what you end up getting, is like the Christmas icicle lights. You have the main strip, which is the array, and then linked nodes cascading down from each index, the icicles.

In Hash maps we have to understand what a node is. So, we created a child class to define what it meant to be a node. Which is essentially in separate chaining. Once we defined what a node is, we began filling in the methods. Put worked just like before, except for instead of hashing it a second time when an index was already full, it just linked the newly created node to the node that was already in the index. The get method worked similar as well, except for this time we were just comparing nodes. And, we would just cascade down the linked list of nodes at the specific index instead of the entire array. Since, Hash Maps can have multiple values at one index, we couldn’t just count how many values there were. Instead, every time we put an element, we increment size and then we just return that size value for our size method. And was again, our toString method just goes though all indexes. If at any index there is more than one node attached, it will go down the chain. It is just meant to print off everything neatly.

Here are my results:

All the tests passing:



The profiler produced these results:

