**Assignment 6**

**SWE 437**

**Spring 2018**

**Team Members: Ahmad Aram, Khang Chau Vo and Ranjit Singh**

**Source Code:** [**https://github.com/davey108/SWE-437-Quotes-Server**](https://github.com/davey108/SWE-437-Quotes-Server)

The Quotes application story summary involves the user opening the application and being presented with the main menu, from here the user may select to search by keyword. This will allow them to type in a keyword that is listed and search for any quotes tagged with that keyword. It will then display the results on the screen for the user.

**User Story:**

“As a user, I will be presented with a list of keywords that I can look for quotes associated with those keywords. Then, I will be able to type in a keyword and the application will display a list of quotes that matches the keyword on my screen”

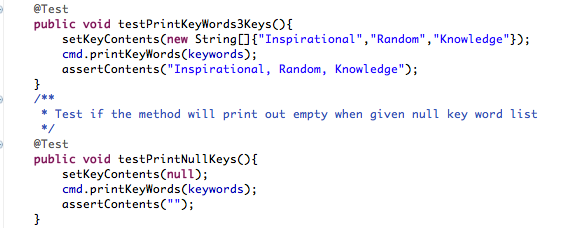
**Our Process:**

* We split the whole user story into 3 components:

1. Print out the list of keywords
2. Check user input for a valid keyword
3. Output a list of quotes based on the user’s keyword

**Component 1:** Printing out keywords

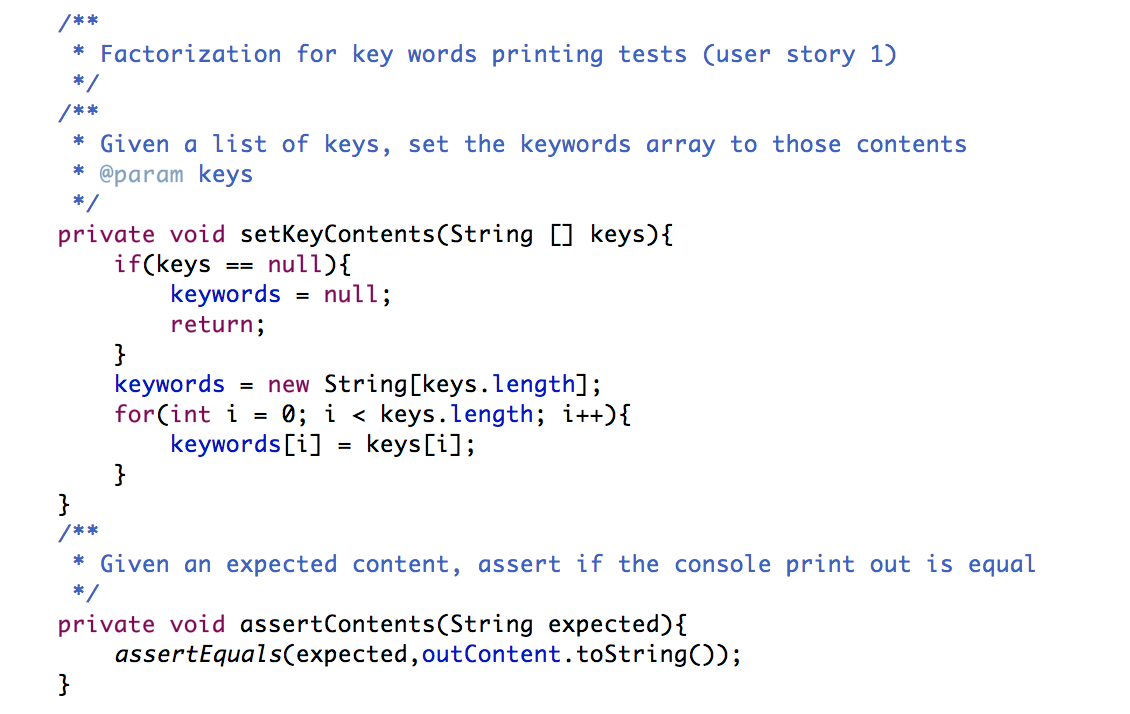
This portion of test verifies that the keywords are printed properly for the user to see upon selection of that feature, below is two of the many tests.

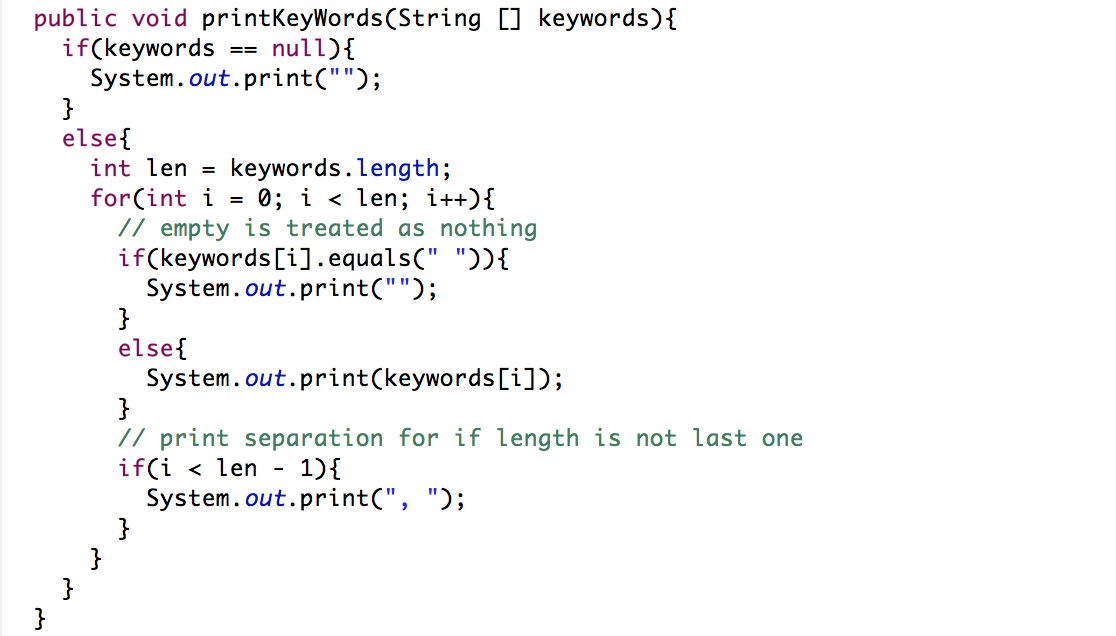


To pass the test, we implemented the method printKeyWords() in the QuoteCMD.java which is responsible for printing a list of keywords to the console given the String [] keywords. At first, we just implemented with a loop to loop through the array and print but later added null case to account for the null keys case.

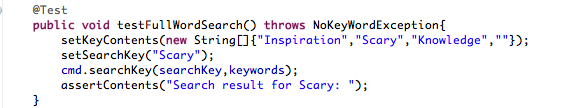
**Component 1 Refactorization/Improvement:**

Originally, the test cases has the flat implementation of just setting up the array and assertEqual inside each test methods, however, in the refactorization, we made a method to set the content of the array and another method to assertEqual given the expected content. This allows the test code to be more readable and understandable.

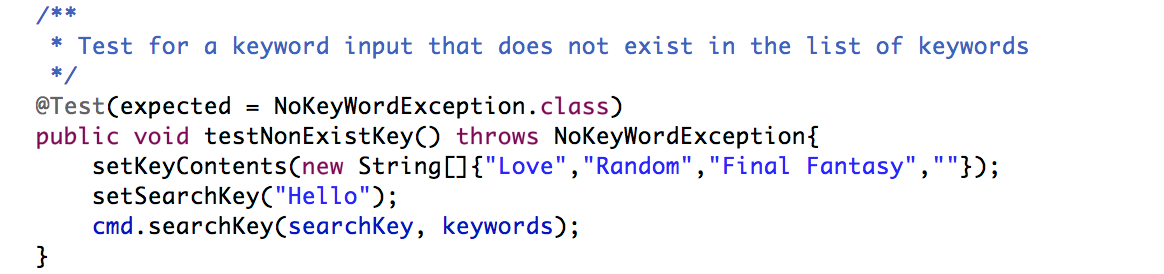


**Final Code for** printKeyWords()

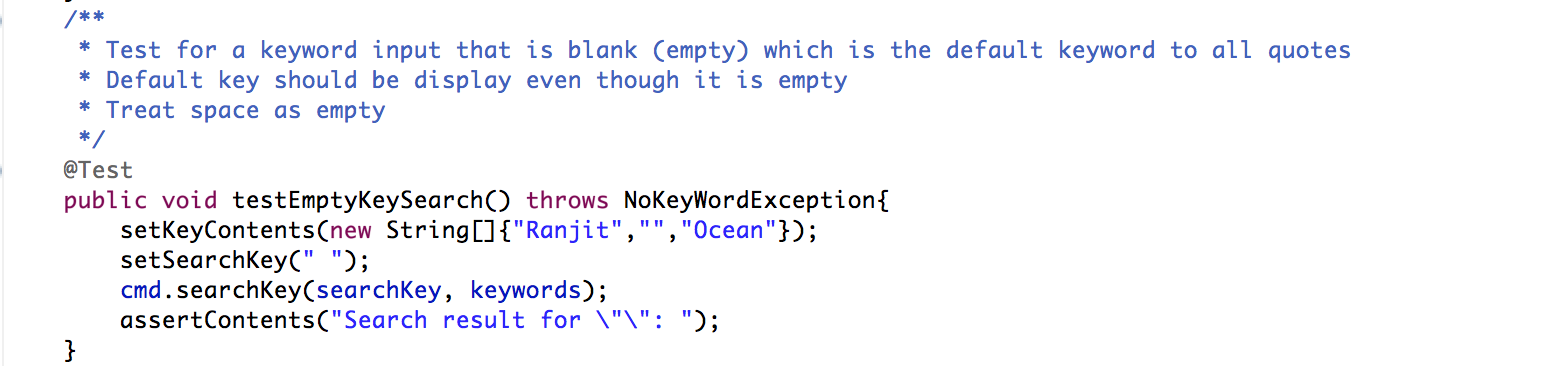
**Component 2:** Checking the user keyword input



This test method was designed to test for an inputted keyword that existed in the list of keyword. We want to make sure that the printout would be correct for that keyword (and the printing list of keyword will be done later). The next method written to meet this test would be searchKey() and basically just search to see if that key exist and print out the above expected message then return null (currently).



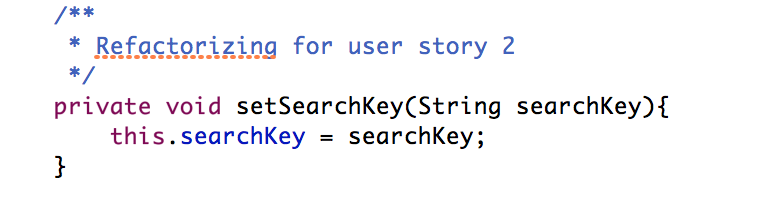
This test method was designed to fail if a key that does not exist in the list is inputted. Instead of returning false, we changed it so that the method would throw a custom NoKeyWordException.



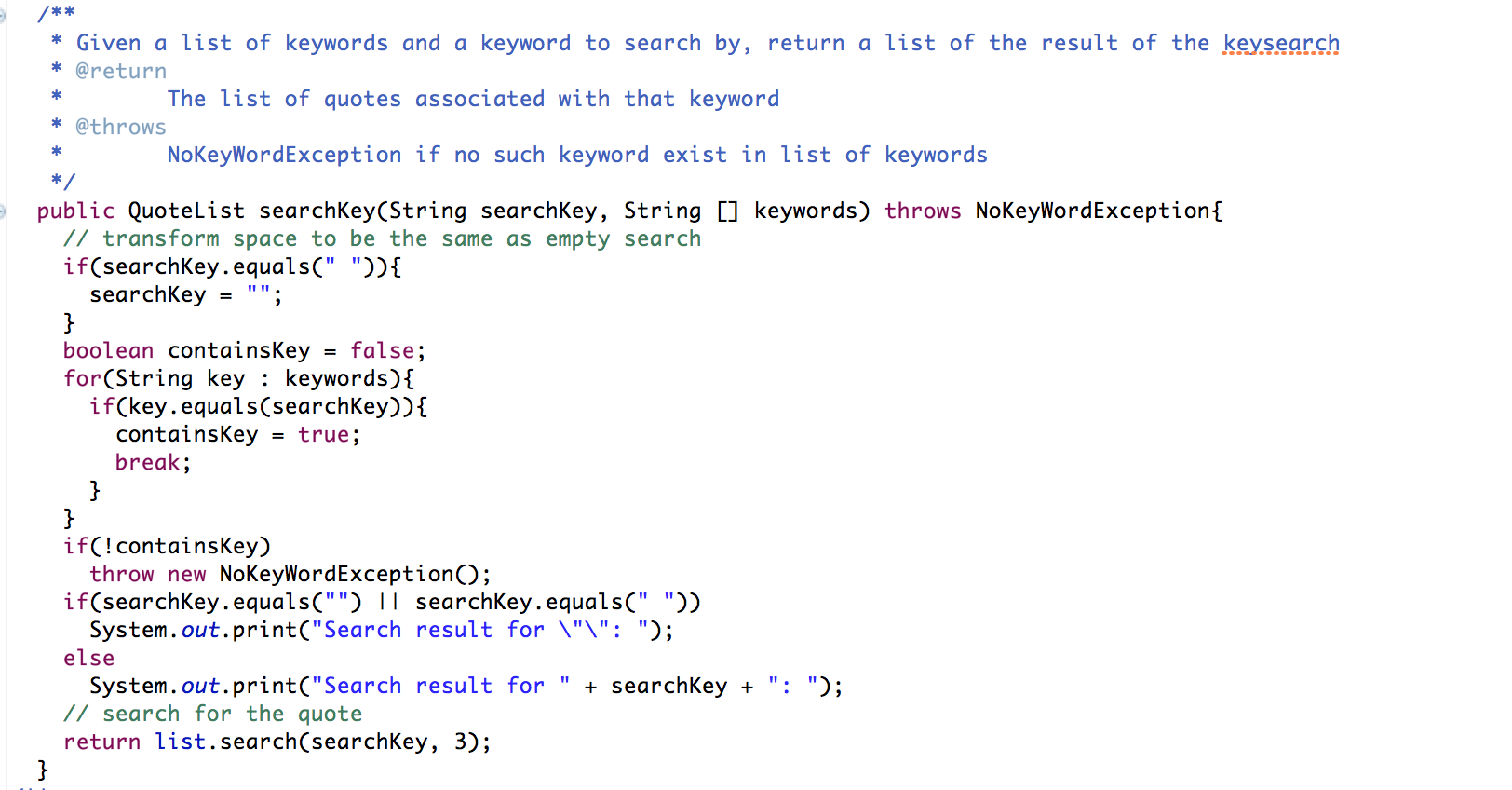
This test method was designed to judge “ “ the same input as “” so we had to remodify searchKey() to change if “ “ was inputted to “” and changed to explicitly print out the “” in the search result message

**Component 2 Refactorization:**

After all the test cases, we realized we were setting the string contents for the searchKey directly so it might be unclear. So we ended up writing a method to set the search key and declared the search key as a global variable. We want to make sure the test methods are clear even though the refactorization might be minimal



**Final Code for** searchKey():

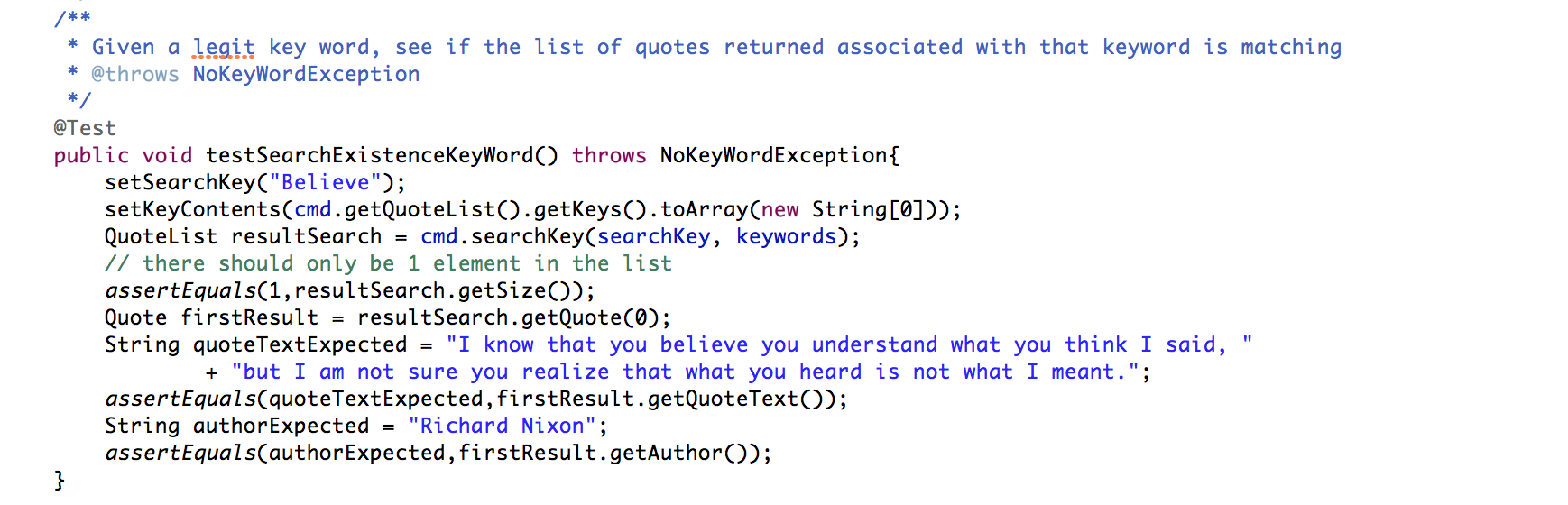


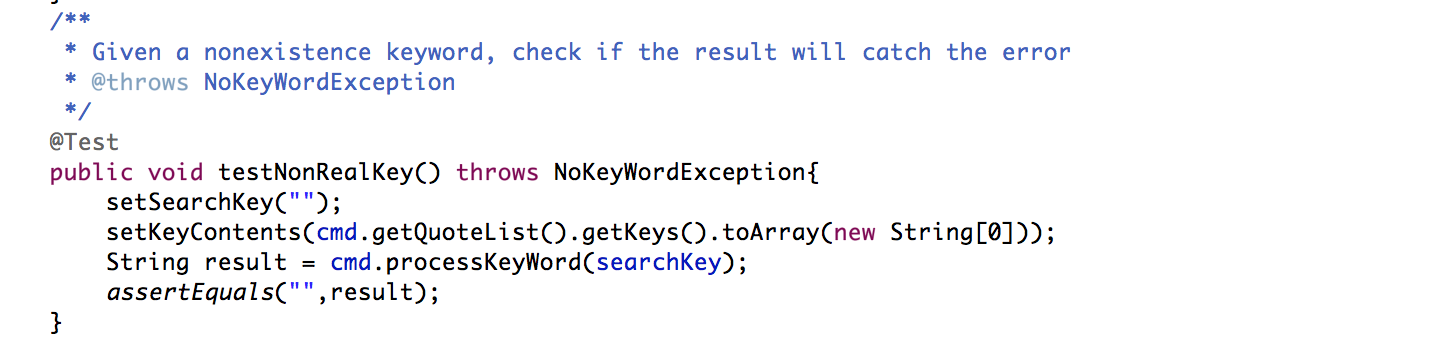
**The method at the time of implementation did not return a valid list, it returned null.**

**Component 3:** Print out list of quotes based on search (or special message if error was thrown from searchKey()

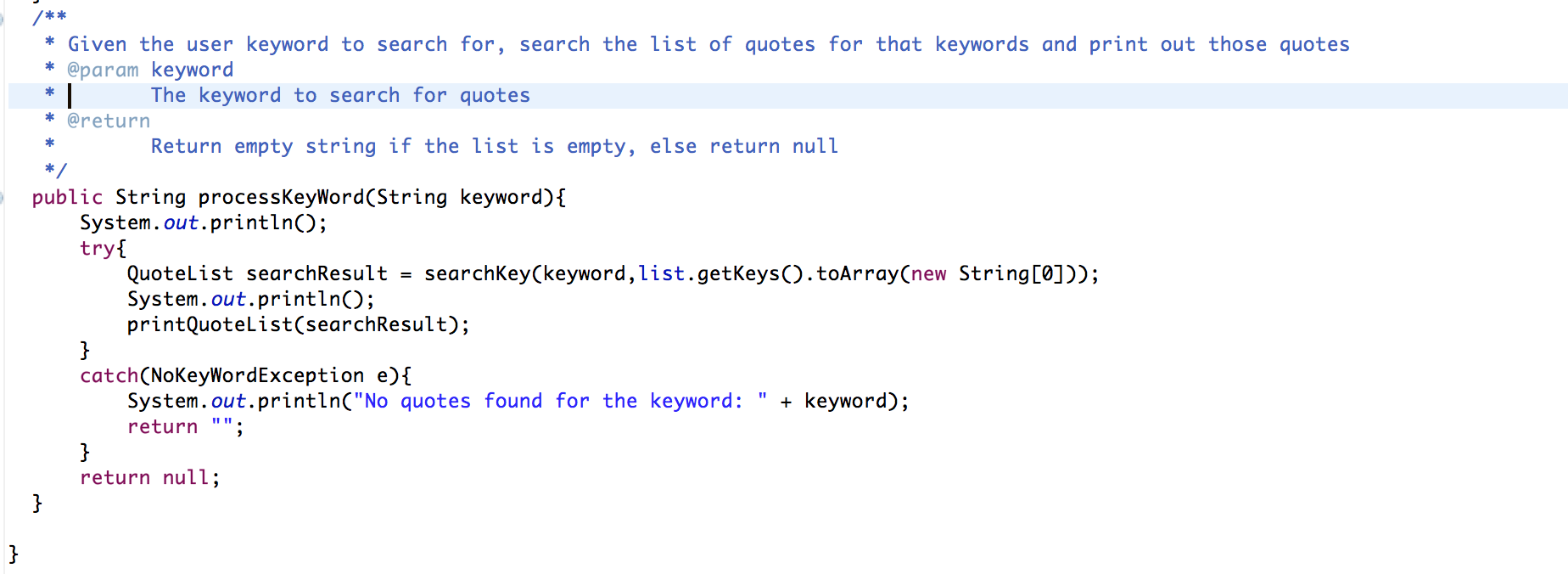
In the last component we added ProcesskeyWord() which allows the program to take in the users search query and searches for a list of quotes that contains that keyword tag and prints out those quotes that are associated with that tag.

The test cases below extend upon those for searchKey() to return the list of quotes given the keyword, we ended up adding a tag category in the XML for all the quotes, and assigning each quote a tag/keyword. We also changed the SaxHandler to process the new tag that was added. As well as the XML Writer to account for the tag/keyword. Changed Quote list search method to allow us to search by a keyword as well.





The NoKeyException isn’t caught in any of the prior methods, so we had to design ProcessKeyword() that is designed to catch that exception and print out a special message if the keyword isn’t valid. In the method we make a call to the searchKey() method and after the search result we print the list of correct Quotes. Below is a image of the processKeyWord() method.



**Screenshots:**

