cse15l-lab-reports

Lab Report 2

Simpleist Search Engine

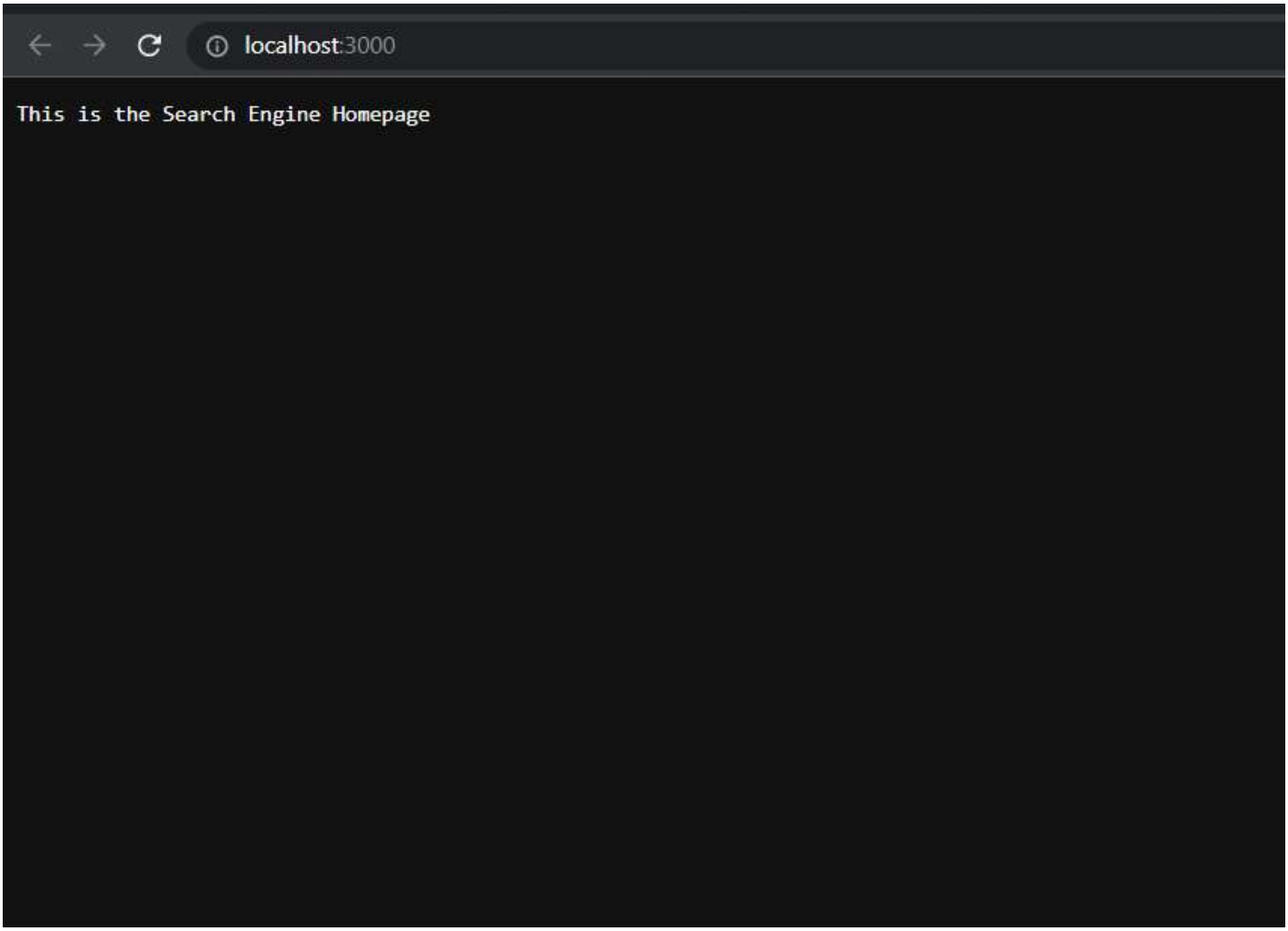
Here is my code for the Simpleist Search Engine:

```
import java.io.IOException;
import java.net.URI;
import java.util.ArrayList;

class Handler implements URLHandler {

    int num = 0;
    ArrayList<String> listRequest = new ArrayList<String>();

    public String handleRequest(URI url) {
        String finalOutput = "";
        if (url.getPath().equals("/")) {
            return String.format("This is the Search Engine Homepage");
        }
        else if (url.getPath().equals("/add")) {
            String[] parameters = url.getQuery().split("=");
            if (parameters[0].equals("s")) {
                listRequest.add(parameters[1]);
                return String.format("Adding... ", listRequest.get(listRequest.size() - 1 ));
                }
            }
        else if (url.getPath().equals("/search")) {
            String[] parameters = url.getQuery().split("=");
                if (parameters[0].equals("s")) {
                    for(String s: listRequest){
                        if(s.contains(parameters[1])){
                            finalOutput = finalOutput + " " + s;
                        }
                    }
                    return finalOutput;
                }
            }
            return String.format("We couldn't quite find what you were looking for.");
```

```
            }
        }

    class SearchEngine {
        public static void main(String[] args) throws IOException {
            if(args.length == 0){
                System.out.println("Missing port number! Try any number between 1024 to 49151");
                return;
            }

            int port = Integer.parseInt(args[0]);

            Server.start(port, new Handler());
        }
    }
```

Homepage:

This homepage is the default for the Search Engine. It calls the `handleRequest` function, which reads the `"/"` in the search bar and then outputs `"This is the Search Engine Homepage"`.



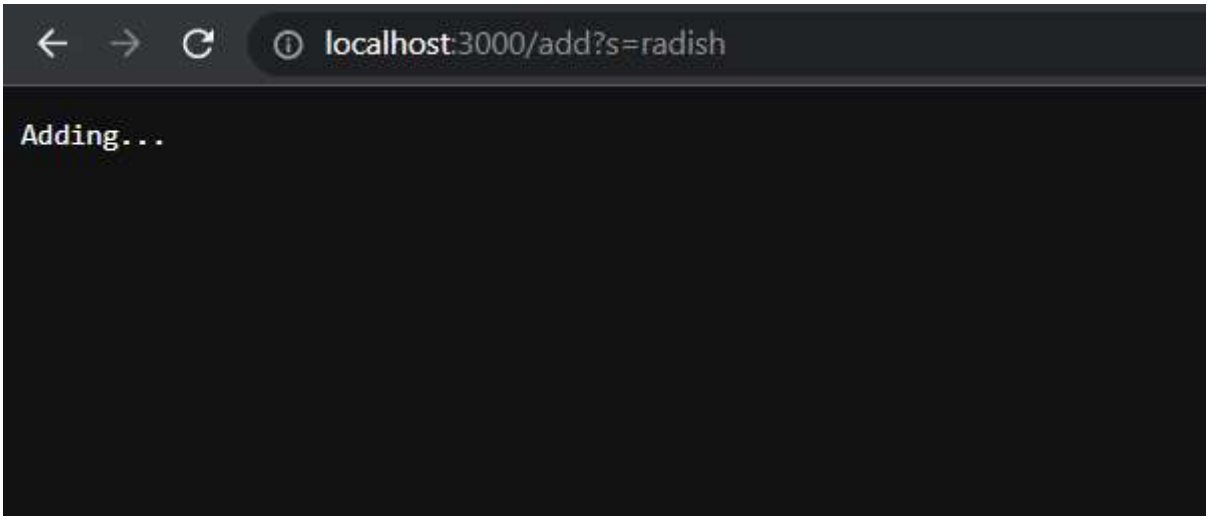
Adding an Element:

This code calls the `handleRequest` method again, except that it reads the `"/add"` in the url, and goes to the first else-if statement. It creates a String array called `parameters` that looks through the url to find the query symbol, `"?"`, and then split the url after the = sign. Then, it will check if `parameters[0]` matches `"s"`, which is an indicator that something should be called.

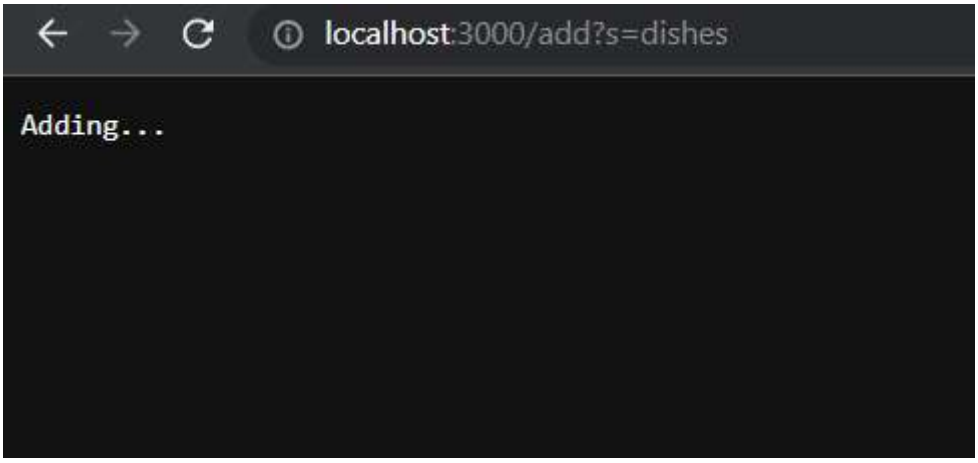
Then, after creating an ArrayList called listRequest, the code calls for `parameters[1]`, or the actual element we want to be added, into the listRequest to be stored.

This function was called three times to add two elements named `"radish"`, and one named `"dishes"`.

Adding element radish:



Adding element dishes:

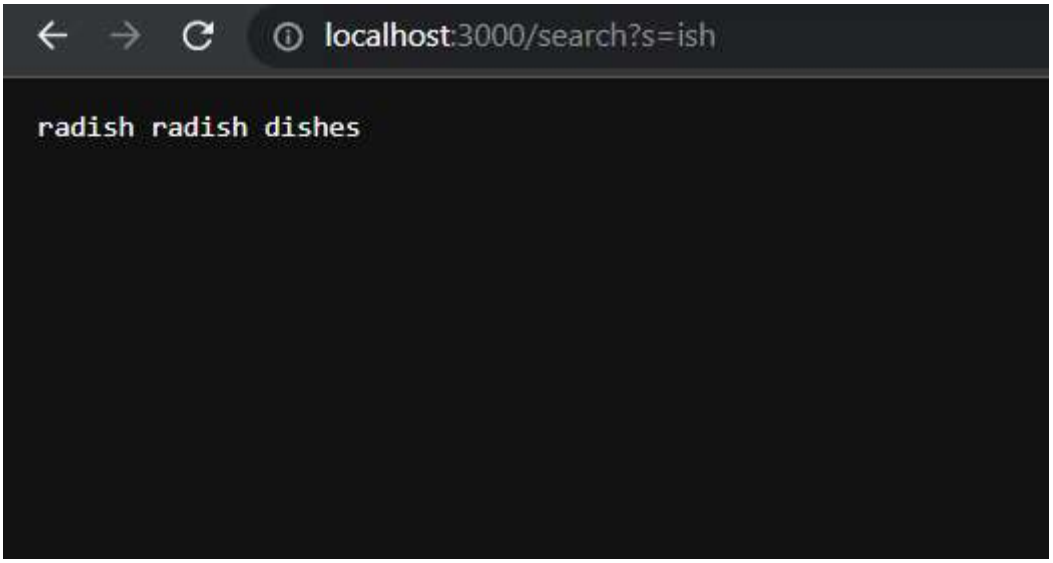


After successfully adding the element in the search bar, the page changes from the homepage to a page that says `"Adding..."`.

Search Query:

The search statement, or the last else-if statement, can only be called successfully after other elements have been added in. It also calls upon the `handleRequest` function. The code creates a new String Array called parameters, which does the same exact thing as the String Array in the `/add` function, except is separately regarded. Once again, it will search the url until it finds the query symbol, and then split it after the = symbol, and then check if `parameter[0]` equals `"s"`.

Next, I created a for loop that will loop through the elements in `listRequest`, which contains all the elements that were added in `/add`, and then check to see if any part of the said element contains the phrase in `parameters[1]` of the `/search` url. If it does contain it, then it will be added to a currently empty string called `finalOutput`, and then after all elements are looped through and accounted for in `finalOutput`, I return `finalOutput`.



Part 2: Finding bugs

1. Array Testing

Failure Inducing Input:

The failure inducing input for reversed() was `{5, 7, 5, 77, 422}` .

```
@Test
public void testReversed2(){
  int[] input2 = {5, 7, 5, 77, 422};
  assertArrayEquals(new int[]{422, 77, 5, 7, 5}, ArrayExamples.reversed(input2));
}
```

Symptom:

The symptom was that the test failed, and I got an assertion error.

```
PS C:\Users\audre\OneDrive\Documents\lab3> java -cp ".;lib/junit-4.13.2.jar;lib/hamcrest-co
re-1.3.jar" org.junit.runner.JUnitCore ArrayTests
        at org.junit.Assert.assertArrayEquals(Assert.java:429)
        at ArrayTests.testReversed2(ArrayTests.java:25)
        ... 30 trimmed
Caused by: java.lang.AssertionError: expected:<422> but was:<0>
        at org.junit.Assert.fail(Assert.java:89)
        at org.junit.Assert.failNotEquals(Assert.java:835)
        at org.junit.Assert.assertEquals(Assert.java:120)
        at org.junit.Assert.assertEquals(Assert.java:146)
        at org.junit.internal.ExactComparisonCriteria.assertElementsEqual(ExactComparisonCr
iteria.java:8)
        at org.junit.internal.ComparisonCriteria.arrayEquals(ComparisonCriteria.java:76)
        ... 36 more

FAILURES!!!
Tests run: 4,  Failures: 1

PS C:\Users\audre\OneDrive\Documents\lab3>
```

Fixing the Bug:

To fix the bug, I set the values of newArr on the left, so something like `newArr[i] = arr[arr.length - i - 1];` , then return `newArr` instead of `arr`

```
15    // order
16    static int[] reversed(int[] arr) {
17      int[] newArray = new int[arr.length];
18      for(int i = 0; i < arr.length; i += 1) {
19        newArray[i] = arr[arr.length - i - 1];
20      }
21      return newArray;
22    }
23
```

Relationship Between Symptom and Bug

This was because the code was setting `arr[i]` to the values in `newArr` , instead of assigning `arr's` values to `newArr` . So, it should actually be assigning 0 to the array , because all the values in the empty `newArr` would be 0.

1. List Testing

Failure Inducing Input:

The failure inducing input for filter() was `{"sad", "cry", "food"}` .

```java
public class ListTests {
    @Test
    public void testFilter(){

        List<String> input = new ArrayList<String>();
        input.add(e: "sad");
        input.add(e: "cry");
        input.add(e: "food");

        List<String> expect = new ArrayList<String>();
        expect.add(e: "sad");
        expect.add(e: "food");

        StringChecker sc = new VowelChecker();
        assertEquals(expect,  ListExamples.filter(input, sc));
    }
}
```

Symptom:

The symptom was that there was an assertion error for the test.

```
PS C:\Users\audre\OneDrive\Documents\lab3> java -cp ".;lib/junit-4.13.2.jar;lib/hamcrest-co
re-1.3.jar" org.junit.runner.JUnitCore ListTests
JUnit version 4.13.2
.E
Time: 0.029
There was 1 failure:
1) testFilter(ListTests)
java.lang.AssertionError: expected:<[sad, food]> but was:<[food, sad]>
        at org.junit.Assert.fail(Assert.java:89)
        at org.junit.Assert.failNotEquals(Assert.java:835)
        at org.junit.Assert.assertEquals(Assert.java:120)
        at org.junit.Assert.assertEquals(Assert.java:146)
        at ListTests.testFilter(ListTests.java:35)

FAILURES!!!
Tests run: 1,  Failures: 1
```

Fixing the Bug:

To fix the bug, I added the result to the back of `result` instead of at the beginning.

```
static List<String> filter(List<String> list, StringChecker sc) {
  List<String> result = new ArrayList<>();
  for(String s: list) {
    if(sc.checkString(s)) {
      // result.add(0, s);
      result.add(s); //changed
    }
  }
  return result;
```

Relation Between Symptom and Bug:

This was because the result was always added to the front of the array instead of the back, so it would always come back reversed instead of sorted like the function was supposed to do.