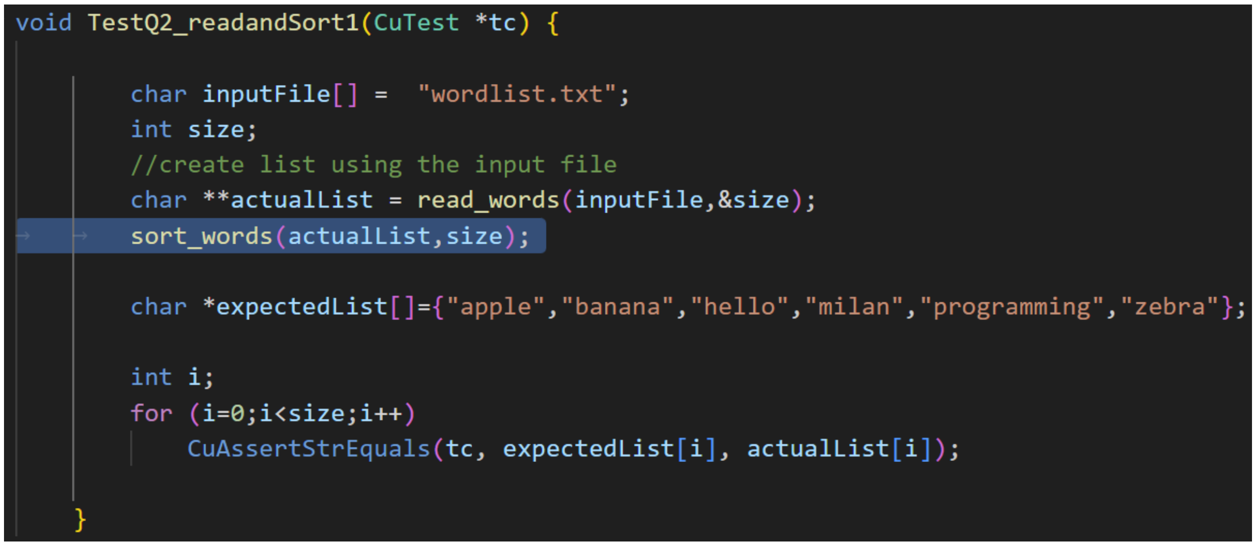
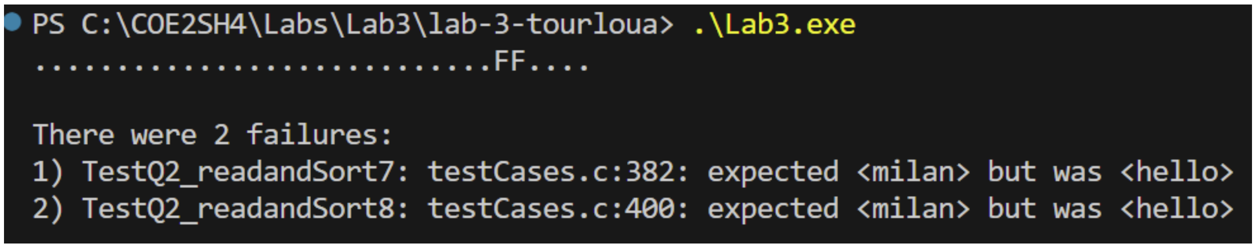
Debug report



```c
void TestQ2_readandSort1(CuTest *tc) {

    char inputFile[] =  "wordlist.txt";
    int size;
    //create list using the input file
    char **actualList = read_words(inputFile,&size);
    sort_words(actualList,size);

    char *expectedList[]={"apple","banana","hello","milan","programming","zebra"};

    int i;
    for (i=0;i<size;i++)
        CuAssertStrEquals(tc, expectedList[i], actualList[i]);

}
```

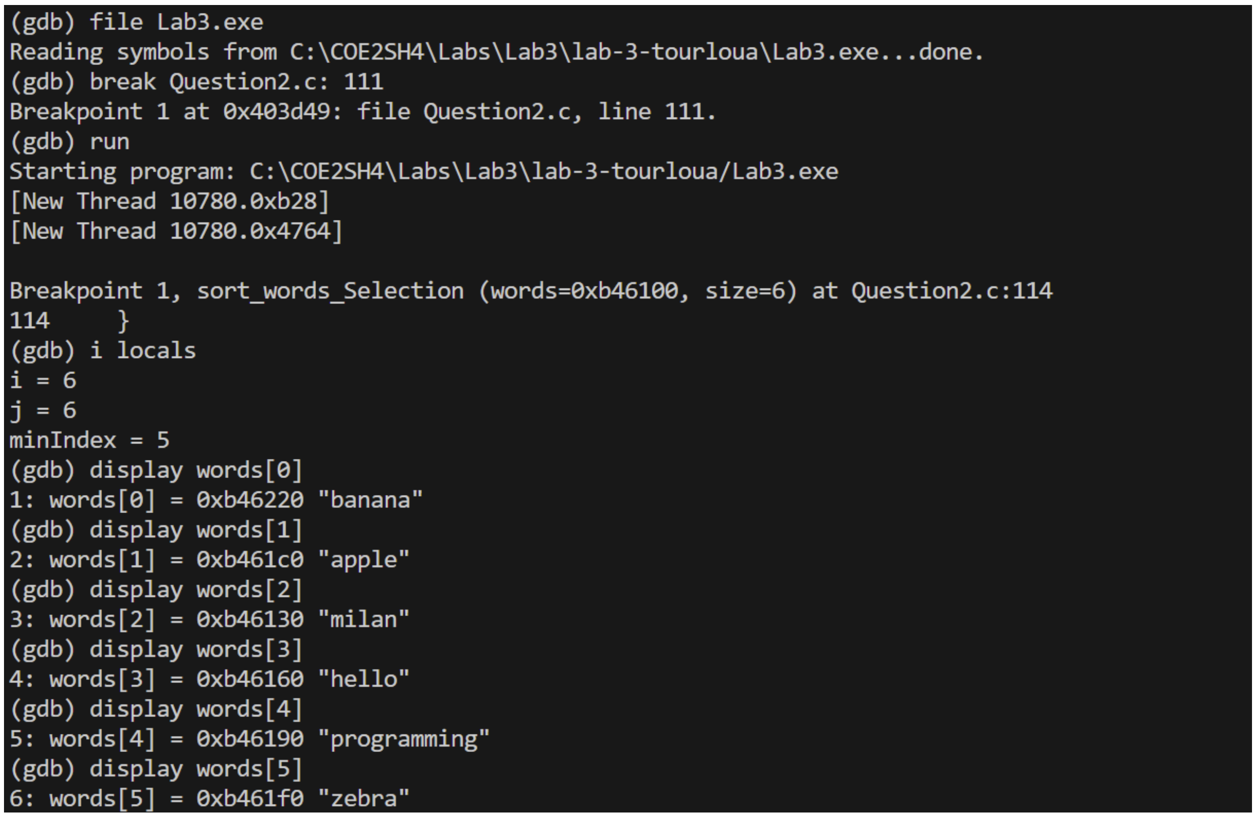
In every question 2 test case, a function called sort_words is called. Nowhere in any of the files does such a function exist. Each of these functions, as well as each additional question 2 test case that followed was renamed to either sort_words_Bubble or sort_words_Selection.



```
PS C:\COE2SH4\Labs\Lab3\lab-3-tourloua> .\Lab3.exe
.............................FF....

There were 2 failures:
1) TestQ2_readandSort7: testCases.c:382: expected <milan> but was <hello>
2) TestQ2_readandSort8: testCases.c:400: expected <milan> but was <hello>
```

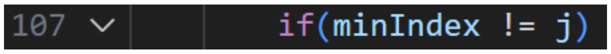
All other test cases were commented out for question 2 functions. The last two test cases are the selection sort test case.

A breakpoint was set up at the end of the function. This was the result.



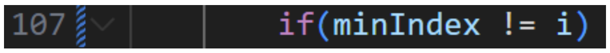
```
(gdb) file Lab3.exe
Reading symbols from C:\COE2SH4\Labs\Lab3\lab-3-tourloua\Lab3.exe...done.
(gdb) break Question2.c: 111
Breakpoint 1 at 0x403d49: file Question2.c, line 111.
(gdb) run
Starting program: C:\COE2SH4\Labs\Lab3\lab-3-tourloua/Lab3.exe
[New Thread 10780.0xb28]
[New Thread 10780.0x4764]

Breakpoint 1, sort_words_Selection (words=0xb46100, size=6) at Question2.c:114
114     }
(gdb) i locals
i = 6
j = 6
minIndex = 5
(gdb) display words[0]
1: words[0] = 0xb46220 "banana"
(gdb) display words[1]
2: words[1] = 0xb461c0 "apple"
(gdb) display words[2]
3: words[2] = 0xb46130 "milan"
(gdb) display words[3]
4: words[3] = 0xb46160 "hello"
(gdb) display words[4]
5: words[4] = 0xb46190 "programming"
(gdb) display words[5]
6: words[5] = 0xb461f0 "zebra"
```

While it does seem that the strings are being partially sorted in ascending order, there is still some way to go.



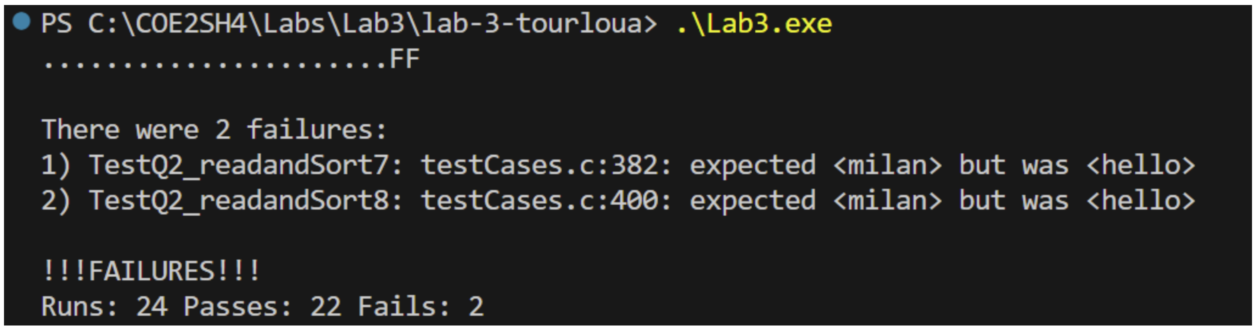
```
        if(minIndex != j)
```

This line was one of the two semantic bugs. minIndex is set to j immediately before this line is run, but only if there is a -ith index string that is greater than a subsequent -jth index string alphabetically. That line was changed to this



```
        if(minIndex != i)
```

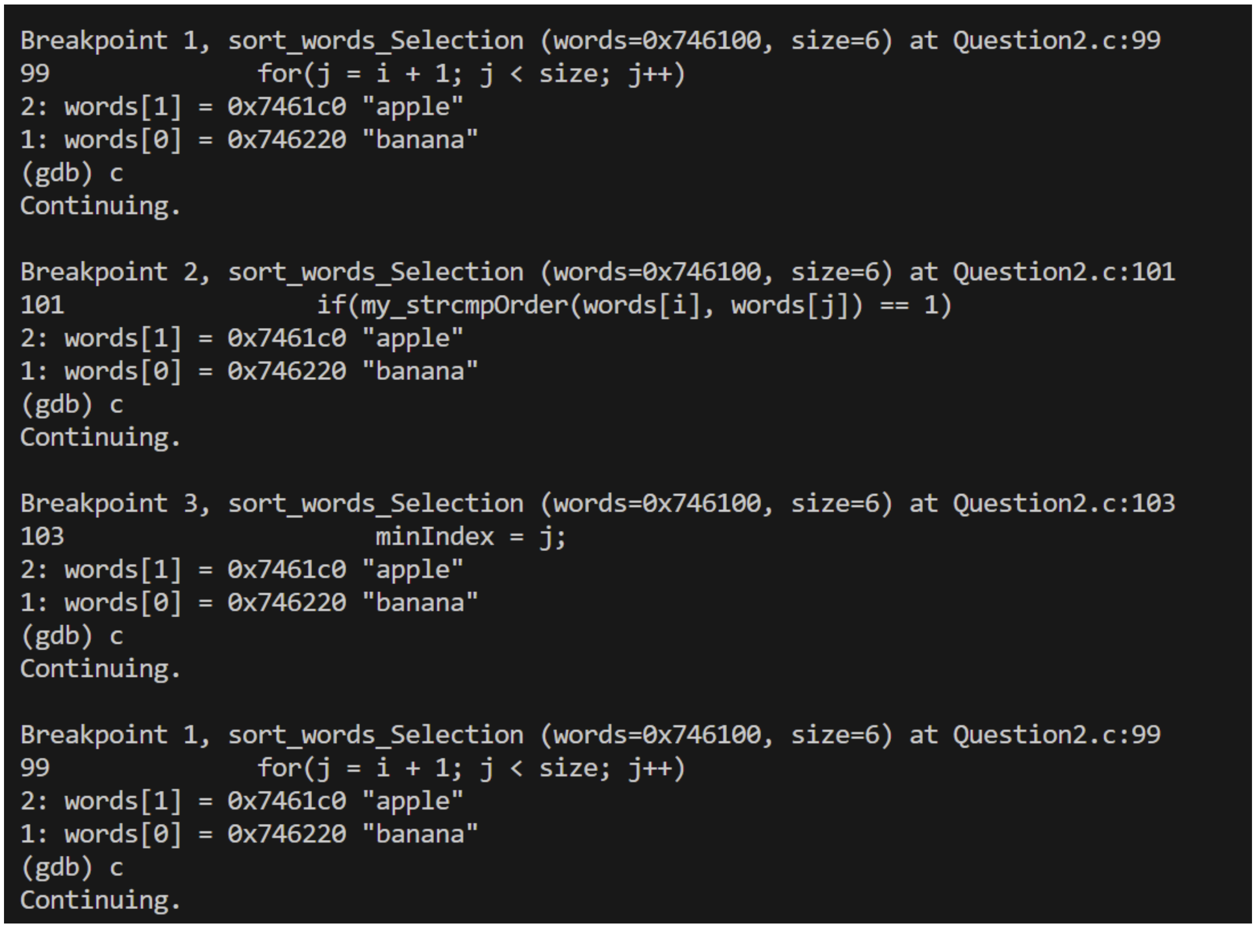
because a swap only occurs if an -ith index is not the smallest string alphabetically. After this change, the program was saved, recompiled and run again. This was the output:



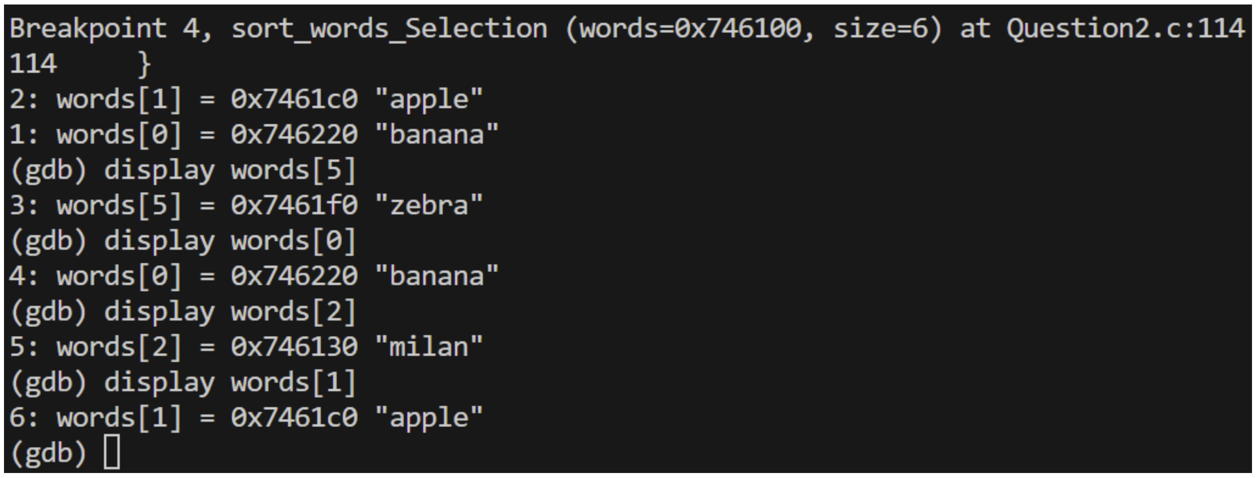
```
PS C:\COE2SH4\Labs\Lab3\lab-3-tourloua> .\Lab3.exe
......................FF

There were 2 failures:
1) TestQ2_readandSort7: testCases.c:382: expected <milan> but was <hello>
2) TestQ2_readandSort8: testCases.c:400: expected <milan> but was <hello>

!!!FAILURES!!!
Runs: 24 Passes: 22 Fails: 2
```

There was still one more semantic error.

After setting up breakpoints throughout the function, in particular at the end of the loops, this was the result:

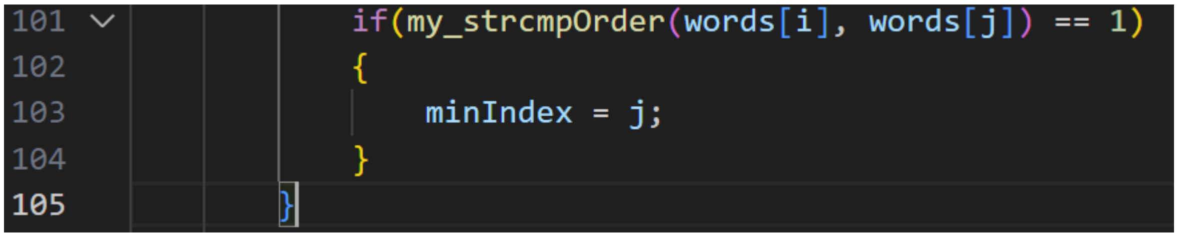


```
Breakpoint 1, sort_words_Selection (words=0x746100, size=6) at Question2.c:99
99              for(j = i + 1; j < size; j++)
2: words[1] = 0x7461c0 "apple"
1: words[0] = 0x746220 "banana"
(gdb) c
Continuing.

Breakpoint 2, sort_words_Selection (words=0x746100, size=6) at Question2.c:101
101                 if(my_strcmpOrder(words[i], words[j]) == 1)
2: words[1] = 0x7461c0 "apple"
1: words[0] = 0x746220 "banana"
(gdb) c
Continuing.

Breakpoint 3, sort_words_Selection (words=0x746100, size=6) at Question2.c:103
103                     minIndex = j;
2: words[1] = 0x7461c0 "apple"
1: words[0] = 0x746220 "banana"
(gdb) c
Continuing.

Breakpoint 1, sort_words_Selection (words=0x746100, size=6) at Question2.c:99
99              for(j = i + 1; j < size; j++)
2: words[1] = 0x7461c0 "apple"
1: words[0] = 0x746220 "banana"
(gdb) c
Continuing.
```



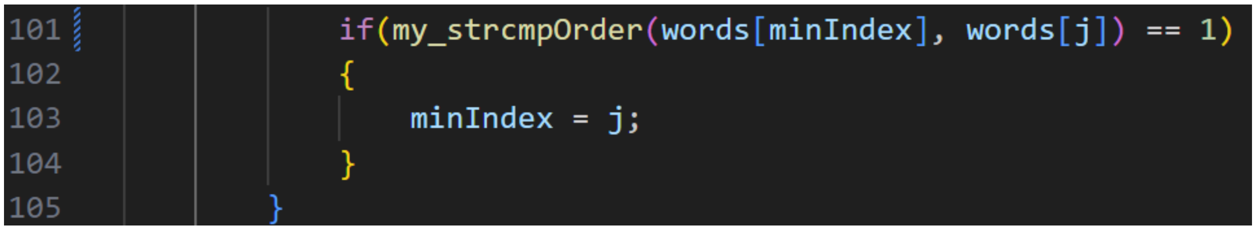
```
Breakpoint 4, sort_words_Selection (words=0x746100, size=6) at Question2.c:114
114     }
2: words[1] = 0x7461c0 "apple"
1: words[0] = 0x746220 "banana"
(gdb) display words[5]
3: words[5] = 0x7461f0 "zebra"
(gdb) display words[0]
4: words[0] = 0x746220 "banana"
(gdb) display words[2]
5: words[2] = 0x746130 "milan"
(gdb) display words[1]
6: words[1] = 0x7461c0 "apple"
(gdb) 
```

The program still was not properly sorting the strings. The problem ended up being here



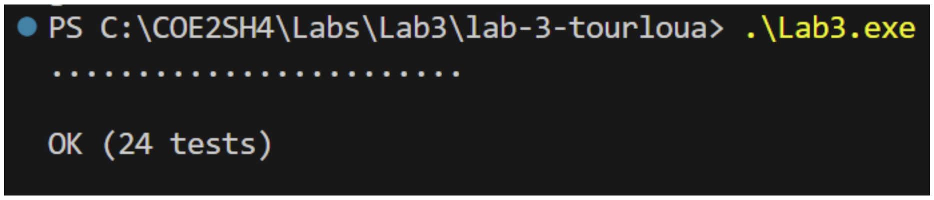
```
101                     if(my_strcmpOrder(words[i], words[j]) == 1)
102                     {
103                         minIndex = j;
104                     }
105                 }
```

Line 101 passes the wrong two strings as parameters. The minIndex string and string j must be compared since we end up swapping the i-th and j-th string later in the function. This code should be replaced with



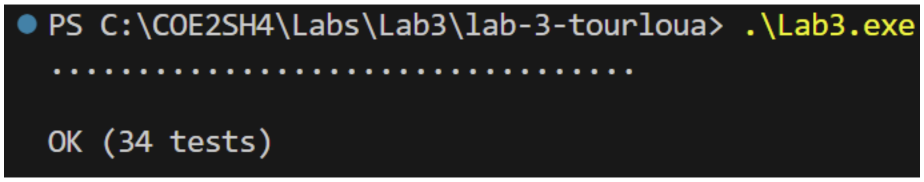
```
101                     if(my_strcmpOrder(words[minIndex], words[j]) == 1)
102                     {
103                         minIndex = j;
104                     }
105                 }
```

With those two semantic bugs removed, as well as the syntax error in the test cases, the program ran successfully.



```
PS C:\COE2SH4\Labs\Lab3\lab-3-tourloua> .\Lab3.exe
........................

OK (24 tests)
```

The program ran successfully with every single test case that was used.



```
PS C:\COE2SH4\Labs\Lab3\lab-3-tourloua> .\Lab3.exe
..................................

OK (34 tests)
```