# Debugging Report – Lab 3

**Vraj Patel**
**400434795**
**Patev61**

## Bug #1:

*Identification of Bug:*

```
119             for(j = i + 1; j < size;
(gdb) i locals
i = 0
j = 5
minIndex = 5
(gdb) s
127             if(minIndex != j)
(gdb)
```

- In the picture above we see the j for loop being executed and the minimum index computed is at 5 (which is 'banana') which is incorrect because we know it is supposed to be index 3 (which is 'apple')
- Later in the function, the swap is implemented in 'banana' and not 'apple' which causes the wrong output

*Proposal of Bug Fix:*

```
    {
        if(my_strcmpOrder(words[i], words[j]) == 1)
        {
            minIndex = j;
```

- Right now the code is being compared with the first element of words with the j element and not actually at the minimum index of the code.
- This means that the minIndex will always stay at 0 (which is the first element) and only compare the first element to every other element.
- To fix this, comparing the minIndex to every other element is crucial.

*Validation of Bug Fix:*

```
    {
        if(my_strcmpOrder(words[minIndex], words[j]) == 1)
        {
            minIndex = j;
        }
    }
```

*Picture of new code*

```
119             for(j = i + 1; j < size; j++)
(gdb) i locals
i = 0
j = 5
minIndex = 3
```

# Debugging Report – Lab 3

**Vraj Patel**
**400434795**
**Patev61**

```
(gdb) p*words@size
$1 = {0x7b5798 "milan", 0x7b57d0 "hello", 0x7b5808 "programming", 0x7b5840 "apple", 0x7b5878 "zebra", 0x7b58b0 "banana"}
(gdb)
```

- In the picture above, after the j for loop is done, we can see that the minIndex is 3 (where 'apple' is) which logically makes sense.

## Bug #2:

*Identification of Bug:*

```
115         for(i = 0; i < size; i++)
(gdb) i locals
i = 2
j = 6
minIndex = 5
```

- In the picture above, the maximum value of Index can only be 5 but j will be 6 since it j iterates one last time after for loop

```
        if(minIndex != j)
        {
            swap(&words[i], &words[minIndex]);
        }

    }

}
```

- Using the logic above, this if statement will always be true since minIndex will be <=5 and j would always be 6 after the for loop
- This also means that unnecessary swapping will happen since the if statement will always be true

*Proposal of Bug Fix:*

```
    if(minIndex != j)
    {
        swap(&words[i], &words[minIndex]);
    }
```

Since j always persists the same value since it is after the for loop, it is wise to switch j with. By doing this the if statement wouldn't always be true which fixes the unnecessary swapping.

# Debugging Report – Lab 3

**Vraj Patel**
**400434795**
**Patev61**

*Validation of Bug Fix:*

```
if(minIndex != i)
{
    swap(&words[i], &words[minIndex]);
}
```

*New Code after Bug Fix*

```
(gdb) s
127             if(minIndex != i)
(gdb) i locals
i = 3
j = 6
minIndex = 3
(gdb) s
115         for(i = 0; i < size; i++)
(gdb)
```

- In the code above whenever i and minIndex are equal to each other the if statement isn't conducted and the next iteration starts, as intended to.

```
(gdb) continue
Continuing.
.......................

OK (24 tests)

[Inferior 1 (process 18152) exited normally]
(gdb)
```

The code now runs perfectly without any errors or bugs and passes all test cases.