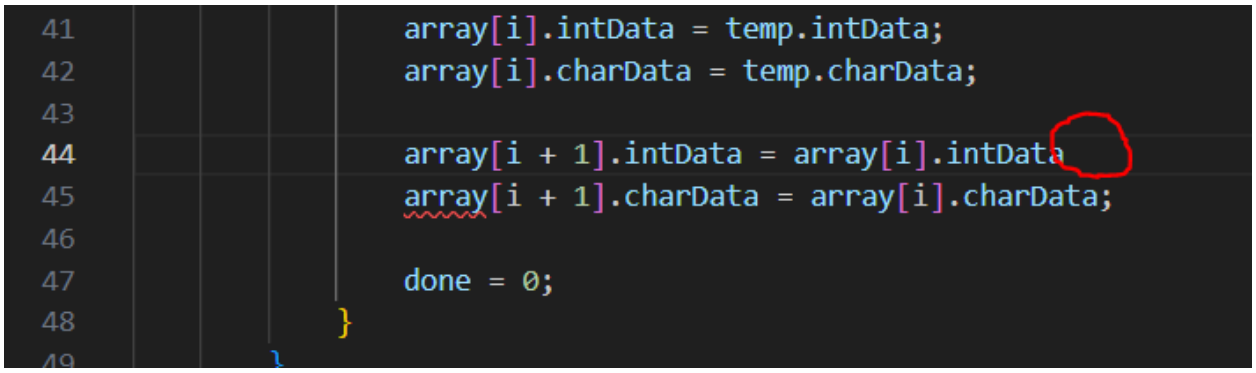
Lab2 Question 4 Debugging Report

Fixing Syntactic Errors:

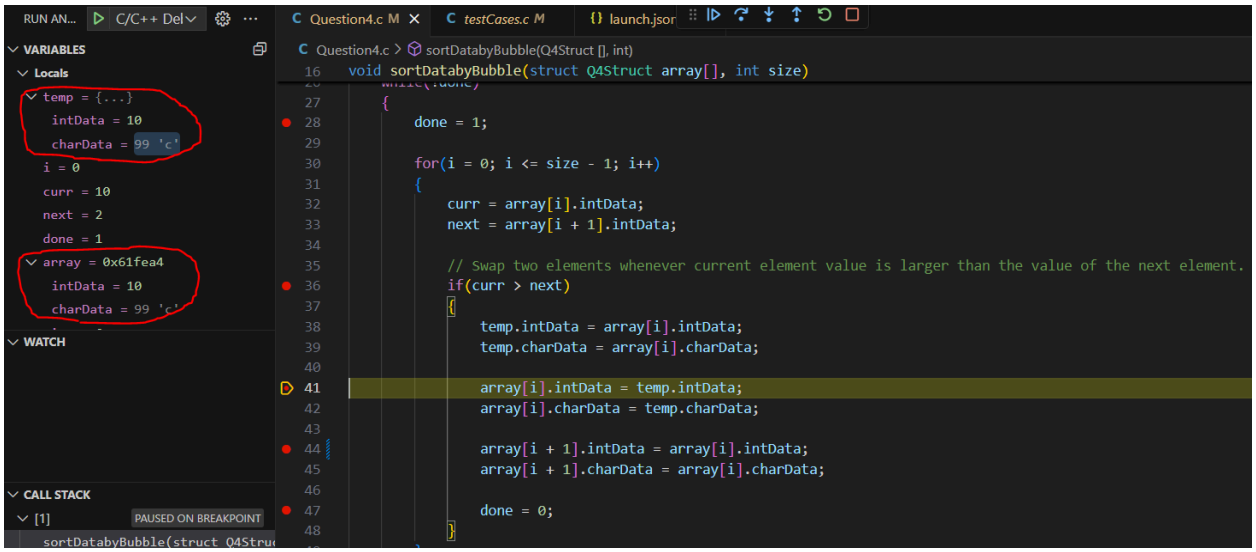


error: expected ';' before 'array'

Fix: Add a semicolon at the end of line 44.

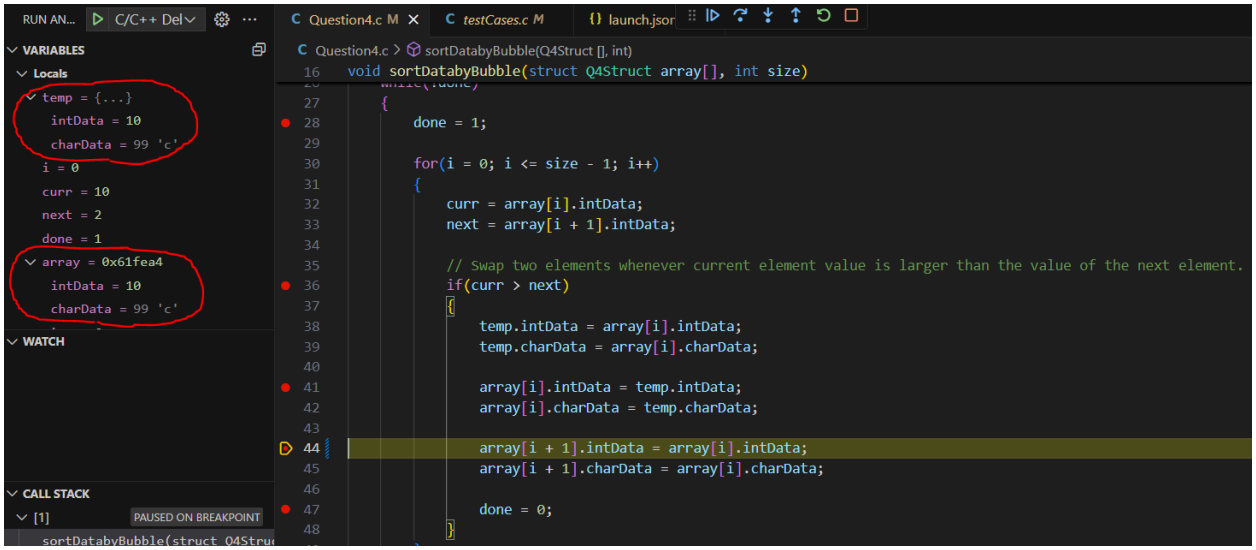
Fixing Semantic Bugs:

Bug #1: Incorrect Swapping Logic



As shown above, lines 38 and 39 have been executed. i = 0, meaning this is the very first pass in the for loop.

Lines 38 and 39 assign the first int and char elements in array[i] to a temp[i] variable, meant to store the values temporarily for swapping.



I've now clicked continue, which executes lines 41 and 42 and stops at the next breakpoint on line 44. Lines 41 and 42 assign the int and char elements in temp[i] to array[i]. However, this is **not correct**. The elements in array[i] were assigned to temp[i] so that array[i + 1] could be assigned to array[i], and then the values in temp[i] could be assigned to array[i + 1]. In other words, this was supposed to happen:

temp[i] = array[i]

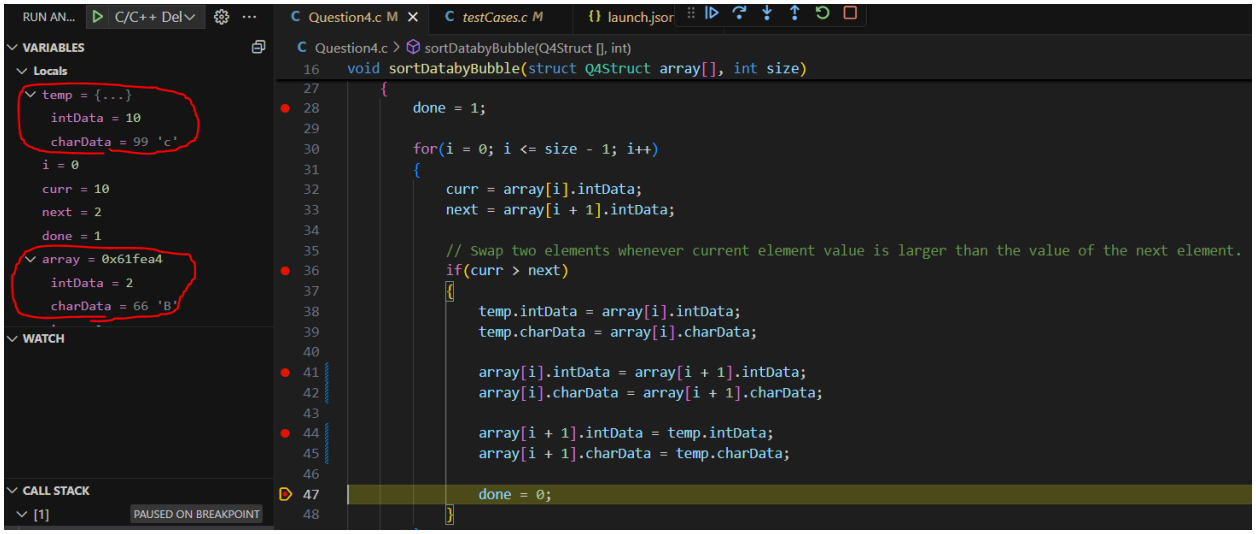
array[i] = array[i + 1]

array[i + 1] = temp[i]

However, that did not happen. This happened:

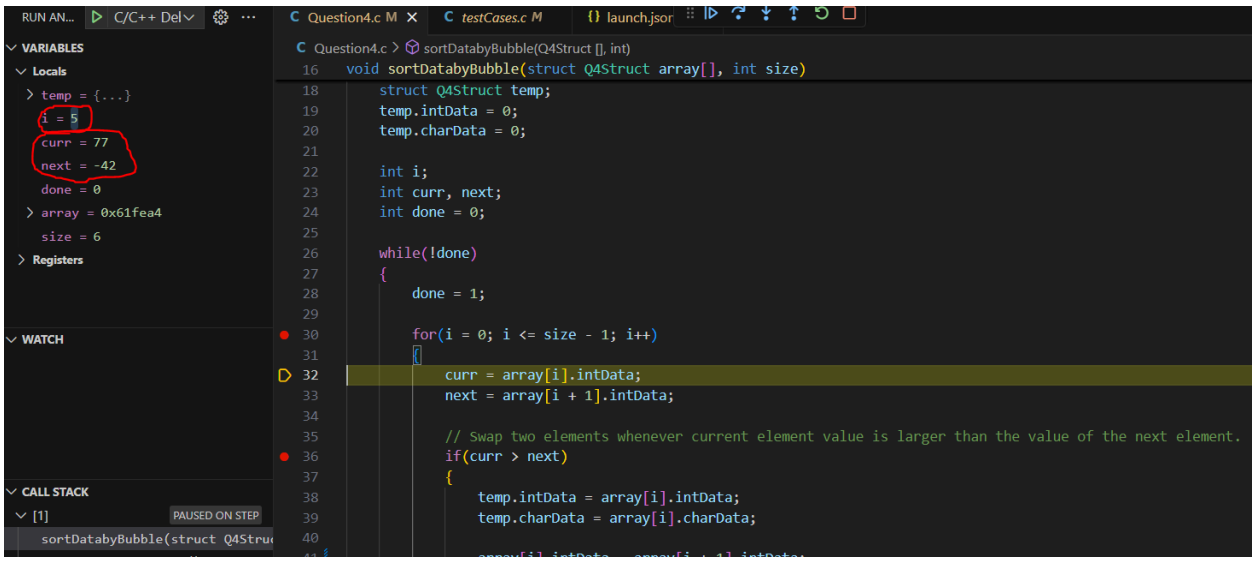
temp[i] = array[i] and then array[i] = temp[i], which achieves nothing, and then array[i + 1] = array[i]. So after all that, temp[i] = array[i] = array[i + 1]. This can be seen in the above image circled in red. The temp values and array values are all the same.

To fix this, we rewrite the code:



As shown above, the swapping logic has been rewritten. Running the debugger again, after 1 pass of the for loop, the elements in array[i] and array[i + 1] have been successfully swapped as temp[i] = array[i + 1] = 10 and ‘c’ and array[i] = 2 and ‘B’.

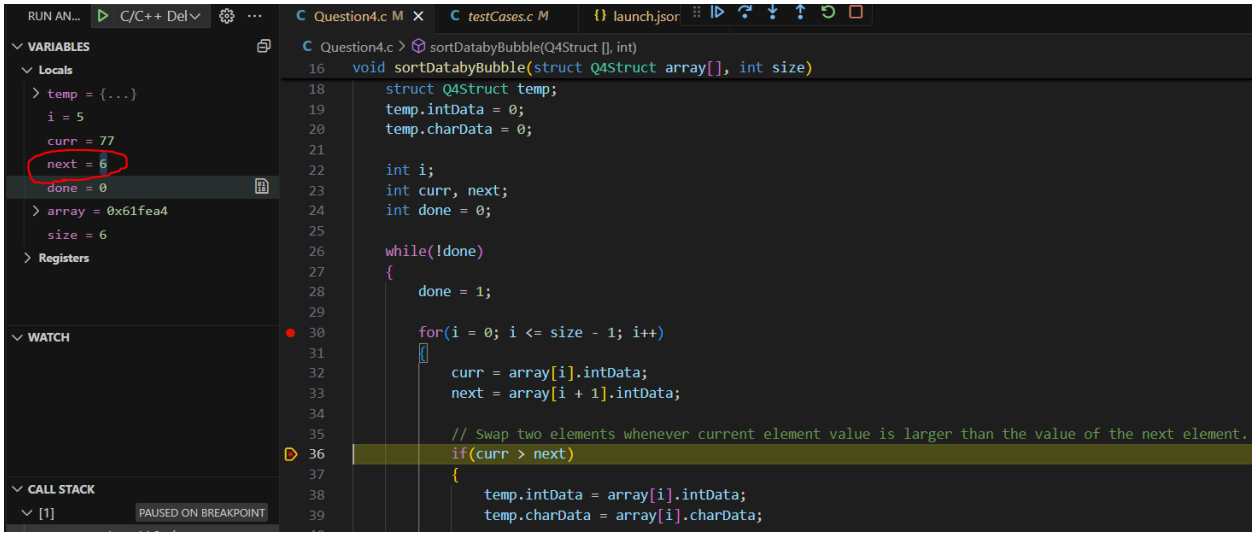
Bug #2: For loop out of bounds



As shown in the screenshot above, i has just been incremented to 5. i = size - 1, so it enters the for loop. Curr and next are still their previous values from i = 4.

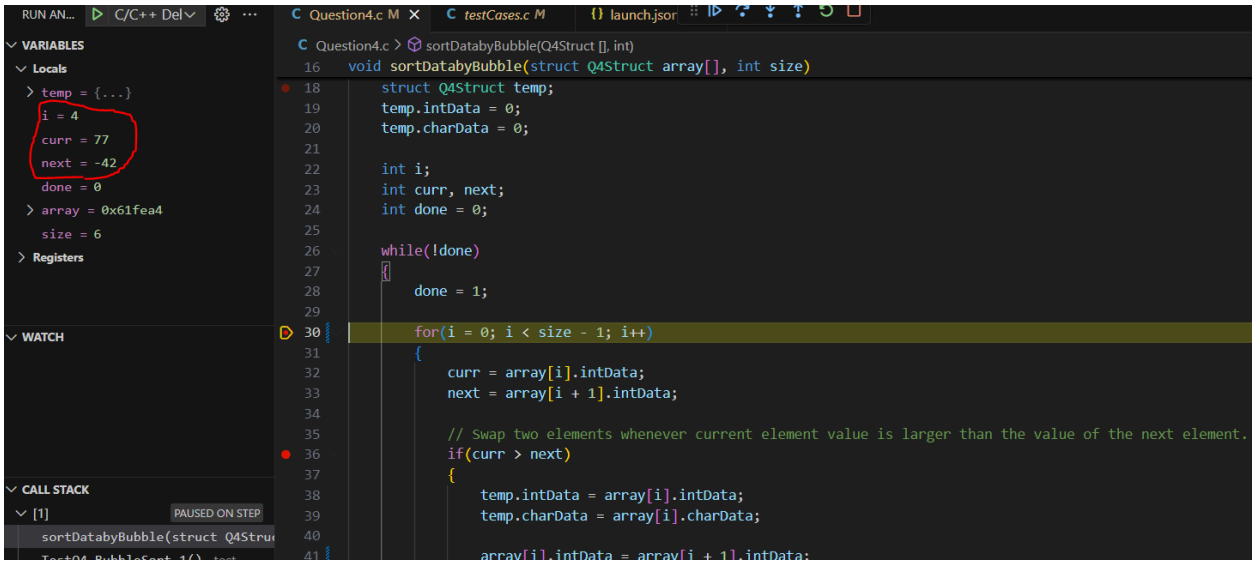
```
struct Q4Struct input[]={{10, 'c'}, {2, 'B'}, {-5, 'k'}, {12, 'z'}, {77, 'a'}, {-42, '?'}};
```

As you can see, those are the last i and i + 1th int values in the array.



Next, lines 32 and 33 are executed. Next is assigned the i + 1th element of the array, which is 5 + 1 = 6. So, it attempts to retrieve array[6], the 7th element of the array, which doesn't exist. A junk value of 6 is then assigned to next, causing a Segmentation error.

To fix this, we simply change i <= size - 1 to i < size - 1. This ensures that the for loop doesn't attempt to retrieve a value from the array that doesn't exist.



This fixes the segmentation fault issue as the final element retrieved in array[] will be array[5], the 6th and final element of the array.