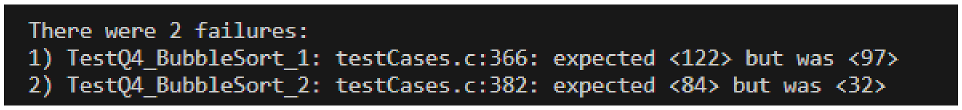
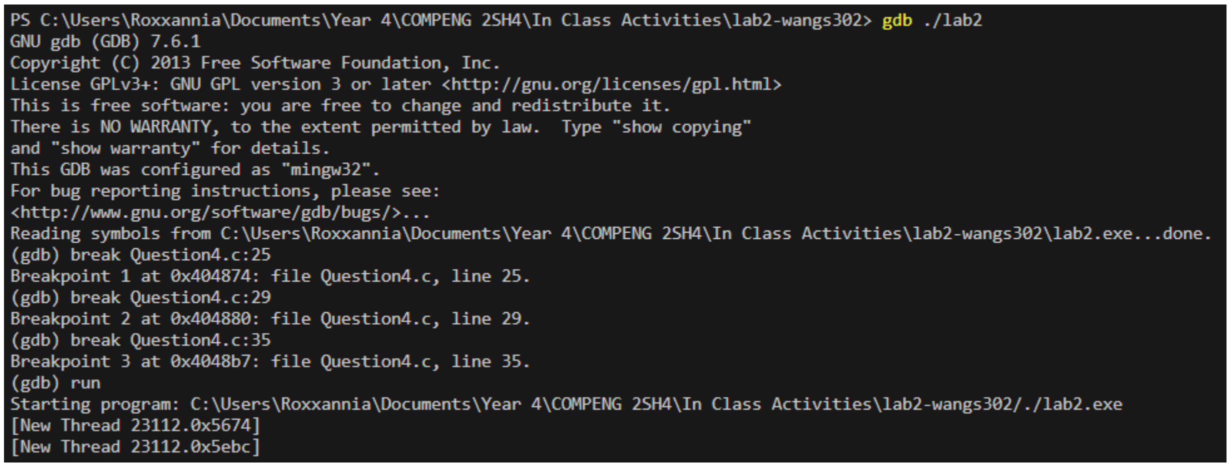
Debugger Report

Prior to fixing the semantic bugs, it appears that the BubbleSort algorithm does not pass its original test cases.



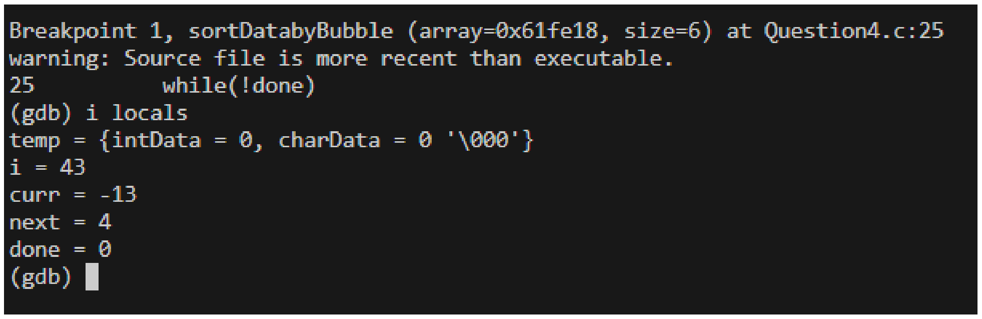
```
There were 2 failures:
1) TestQ4_BubbleSort_1: testCases.c:366: expected <122> but was <97>
2) TestQ4_BubbleSort_2: testCases.c:382: expected <84> but was <32>
```

First bug: Out of Bound access



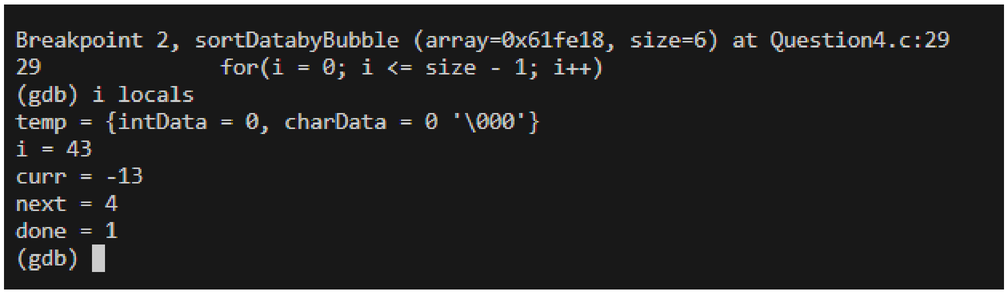
```
PS C:\Users\Roxxannia\Documents\Year 4\COMPENG 2SH4\In Class Activities\lab2-wangs302> gdb ./lab2
GNU gdb (GDB) 7.6.1
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "mingw32".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from C:\Users\Roxxannia\Documents\Year 4\COMPENG 2SH4\In Class Activities\lab2-wangs302\lab2.exe...done.
(gdb) break Question4.c:25
Breakpoint 1 at 0x404874: file Question4.c, line 25.
(gdb) break Question4.c:29
Breakpoint 2 at 0x404880: file Question4.c, line 29.
(gdb) break Question4.c:35
Breakpoint 3 at 0x4048b7: file Question4.c, line 35.
(gdb) run
Starting program: C:\Users\Roxxannia\Documents\Year 4\COMPENG 2SH4\In Class Activities\lab2-wangs302/./lab2.exe
[New Thread 23112.0x5674]
[New Thread 23112.0x5ebc]
```

First, 3 breakpoints is setup at line 25 (while), 29 (for) and 35 (if) to closely examine the behaviour of each conditional statement, then start running the program



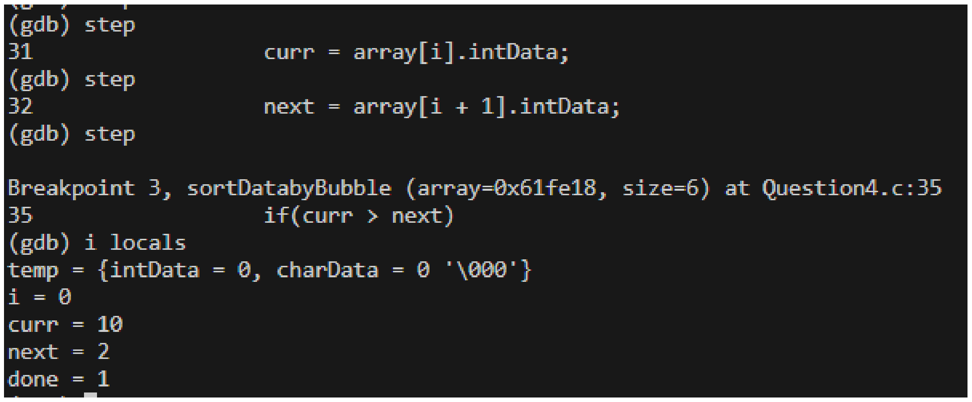
```
Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:25
warning: Source file is more recent than executable.
25          while(!done)
(gdb) i locals
temp = {intData = 0, charData = 0 '\000'}
i = 43
curr = -13
next = 4
done = 0
(gdb)
```

Hitting the first breakpoint. At this point, it has not officially entered the loops yet so the values are randomly assigned other than done = 0;



```
Breakpoint 2, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:29
29                  for(i = 0; i <= size - 1; i++)
(gdb) i locals
temp = {intData = 0, charData = 0 '\000'}
i = 43
curr = -13
next = 4
done = 1
(gdb)
```

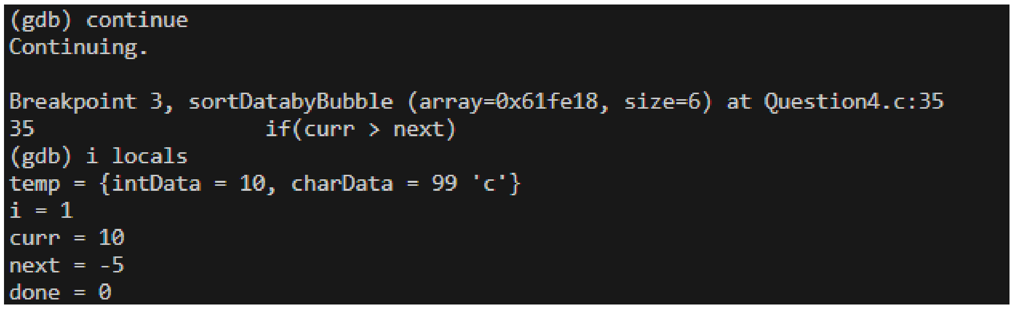
Hitting the second breakpoint, it stops right before the for statement, values are still randomly assigned.



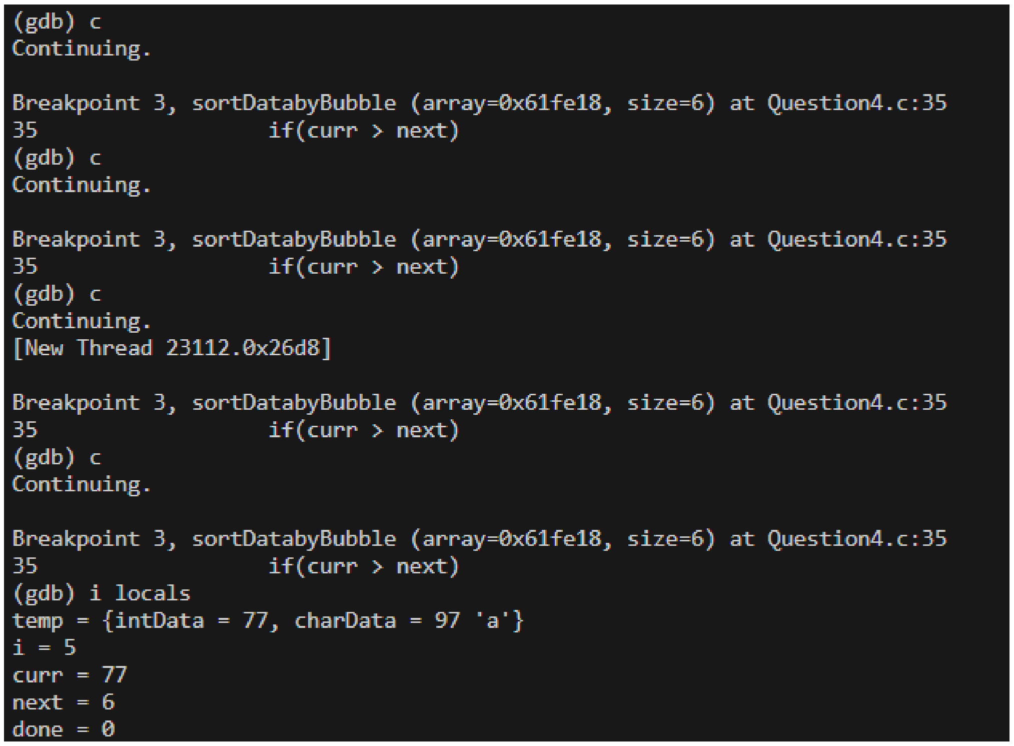
```
(gdb) step
31                  curr = array[i].intData;
(gdb) step
32                  next = array[i + 1].intData;
(gdb) step

Breakpoint 3, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) i locals
temp = {intData = 0, charData = 0 '\000'}
i = 0
curr = 10
next = 2
done = 1
```

Entering the first iteration of first time of running for loop

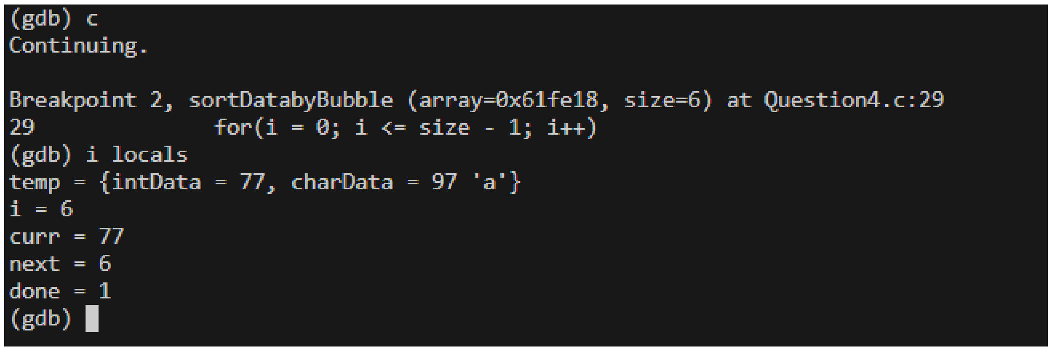
Values appear to be normal



```
(gdb) continue
Continuing.

Breakpoint 3, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) i locals
temp = {intData = 10, charData = 99 'c'}
i = 1
curr = 10
next = -5
done = 0
```



```
(gdb) c
Continuing.

Breakpoint 3, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) c
Continuing.

Breakpoint 3, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) c
Continuing.
[New Thread 23112.0x26d8]

Breakpoint 3, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) c
Continuing.

Breakpoint 3, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) i locals
temp = {intData = 77, charData = 97 'a'}
i = 5
curr = 77
next = 6
done = 0
```



```
(gdb) c
Continuing.

Breakpoint 2, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:29
29              for(i = 0; i <= size - 1; i++)
(gdb) i locals
temp = {intData = 77, charData = 97 'a'}
i = 6
curr = 77
next = 6
done = 1
(gdb) 
```

Continuing the loop until it finishes all iterations. Examining the values by i locals, when the loop is just finished, i = 6 breaks the loop as it reaches the condition for i in the for loop – this means the last value i was properly run in the loop is i = 5 (shown in the figure above). However, if we look at the contents in the loop, every iteration we are accessing array[i] and array[i+1]. When we are in the iteration of i = 5, we are also accessing both array[5] and array[6]. Array[6] is out of bound as the index of the test case array only goes up to 5, resulting in the random number being generated in the 'next = 6' value; there is no '6' in the test case array.

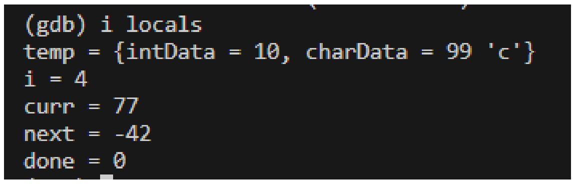
This problem exist in the line:

```
for(i = 0; i <= size - 1; i++)
```

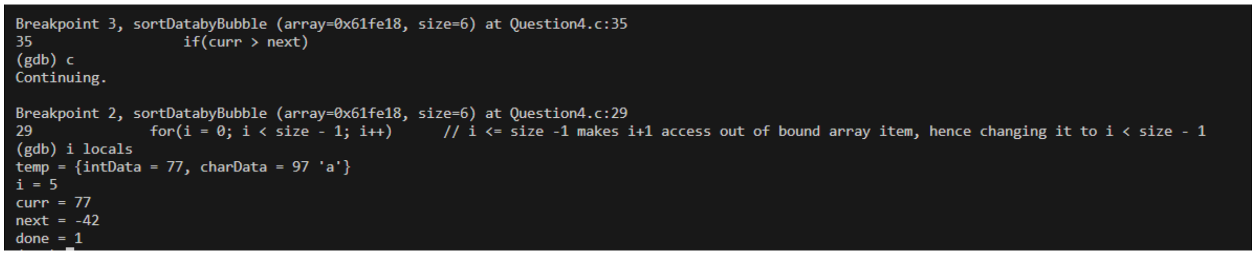
To fix this problem, we need to change the upper bound of i to prevent out of bound access:

```
for(i = 0; i < size - 1; i++)
```

Setting up the same breakpoints and run. Behaviour after fixing out of bound access:



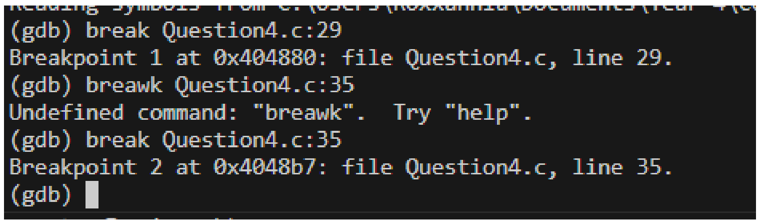
```
(gdb) i locals
temp = {intData = 10, charData = 99 'c'}
i = 4
curr = 77
next = -42
done = 0
```



```
Breakpoint 3, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) c
Continuing.

Breakpoint 2, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:29
29              for(i = 0; i < size - 1; i++)       // i <= size -1 makes i+1 access out of bound array item, hence changing it to i < size - 1
(gdb) i locals
temp = {intData = 77, charData = 97 'a'}
i = 5
curr = 77
next = -42
done = 1
```

We can see that the upper bound of i is limited to 4; when it becomes 5, it breaks out of the loop. As we capped i to 4, when i = 4, array[4+1] stays in bound, access and capture the correct value in the array, and no out of bound values are being accessed or generated. This semantic error is fixed.

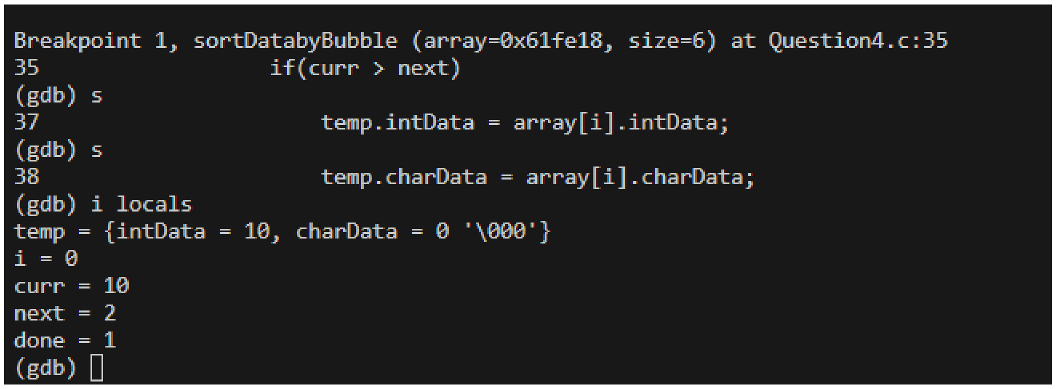
Second bug: Sorting Assignment issue



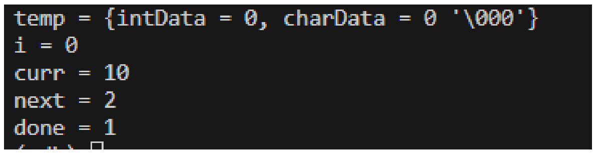
```
(gdb) break Question4.c:29
Breakpoint 1 at 0x404880: file Question4.c, line 29.
(gdb) breawk Question4.c:35
Undefined command: "breawk".  Try "help".
(gdb) break Question4.c:35
Breakpoint 2 at 0x4048b7: file Question4.c, line 35.
(gdb) 
```

Setting up 2 breakpoints at line 29 (for) and line 35 (if)

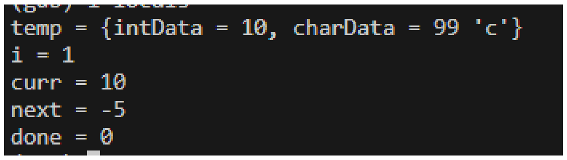
We are going to look at the i locals after each iteration and extends into the first two iteration of the second around of while loop + for loop. We want to look at the variable assigning process in the loop and the conditional statement.



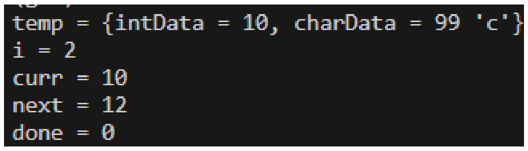
```
Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) s
37                      temp.intData = array[i].intData;
(gdb) s
38                      temp.charData = array[i].charData;
(gdb) i locals
temp = {intData = 10, charData = 0 '\000'}
i = 0
curr = 10
next = 2
done = 1
(gdb) 
```



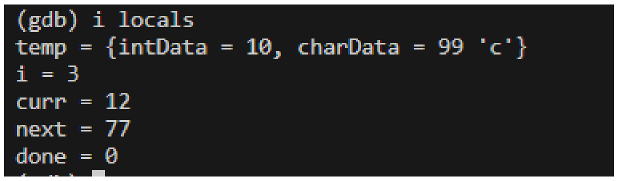
```
temp = {intData = 0, charData = 0 '\000'}
i = 0
curr = 10
next = 2
done = 1
```



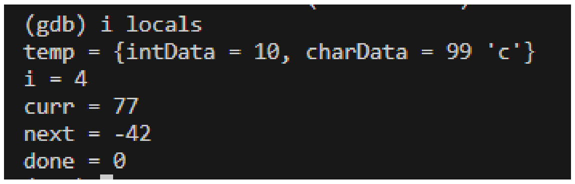
```
temp = {intData = 10, charData = 99 'c'}
i = 1
curr = 10
next = -5
done = 0
```



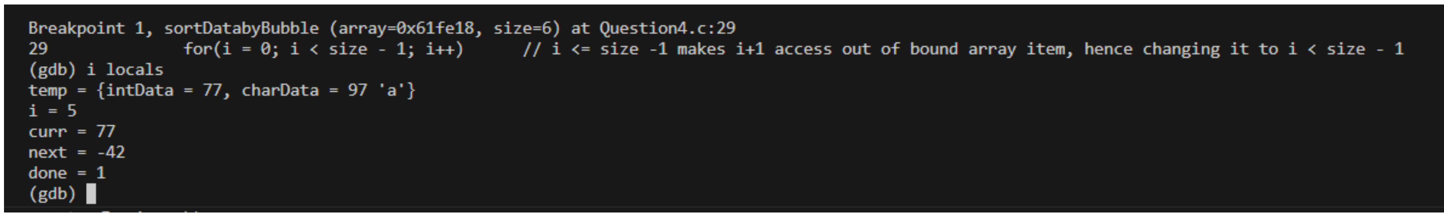
```
temp = {intData = 10, charData = 99 'c'}
i = 2
curr = 10
next = 12
done = 0
```



```
(gdb) i locals
temp = {intData = 10, charData = 99 'c'}
i = 3
curr = 12
next = 77
done = 0
```

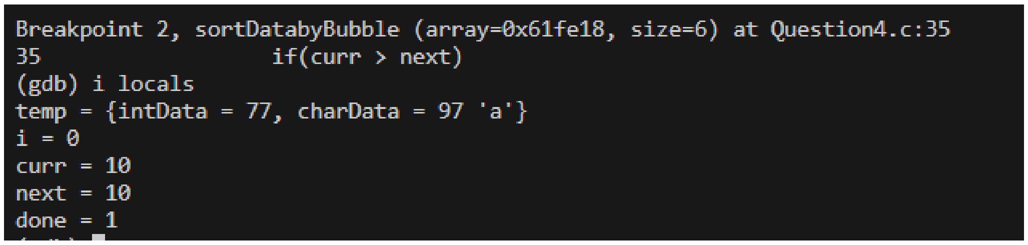


```
(gdb) i locals
temp = {intData = 10, charData = 99 'c'}
i = 4
curr = 77
next = -42
done = 0
```

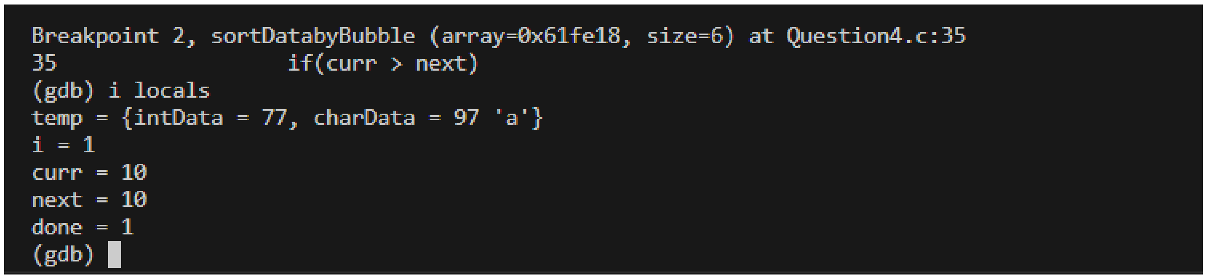


```
Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:29
29              for(i = 0; i < size - 1; i++)      // i <= size -1 makes i+1 access out of bound array item, hence changing it to i < size - 1
(gdb) i locals
temp = {intData = 77, charData = 97 'a'}
i = 5
curr = 77
next = -42
done = 1
(gdb)
```

Second interation as the sort was not completely at the end of the last try



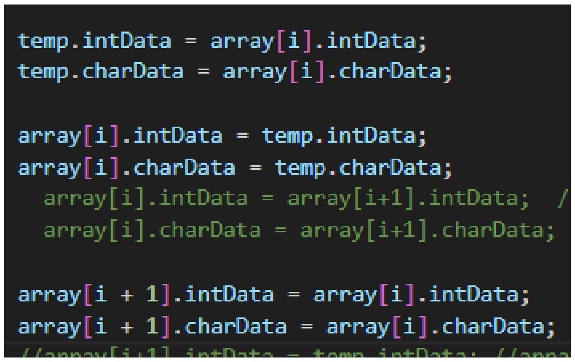
```
Breakpoint 2, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                      if(curr > next)
(gdb) i locals
temp = {intData = 77, charData = 97 'a'}
i = 0
curr = 10
next = 10
done = 1
```



```
Breakpoint 2, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                      if(curr > next)
(gdb) i locals
temp = {intData = 77, charData = 97 'a'}
i = 1
curr = 10
next = 10
done = 1
(gdb)
```

This is no correct, every single entry from index 0 to 3 all become 10, this is not correct.

Looking at the code:



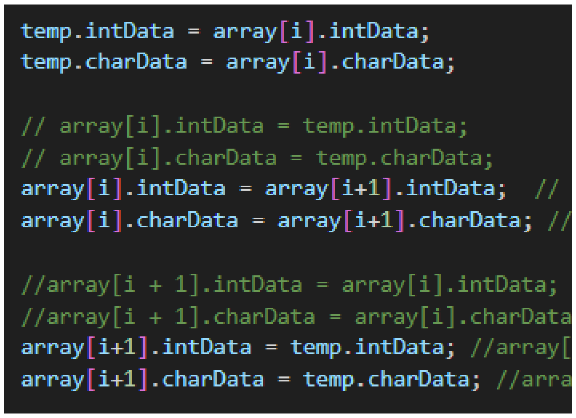
```
temp.intData = array[i].intData;
temp.charData = array[i].charData;

array[i].intData = temp.intData;
array[i].charData = temp.charData;
  array[i].intData = array[i+1].intData;  /
  array[i].charData = array[i+1].charData;

array[i + 1].intData = array[i].intData;
array[i + 1].charData = array[i].charData;
```

Temp was assigned to array[i]

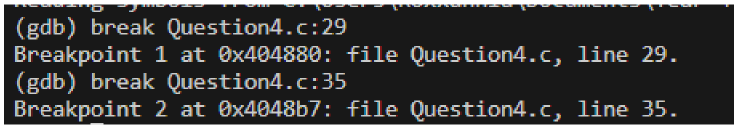
But then array [i] was assigned back to temp data again which was its own values

The same values were assigned to array [i+1]

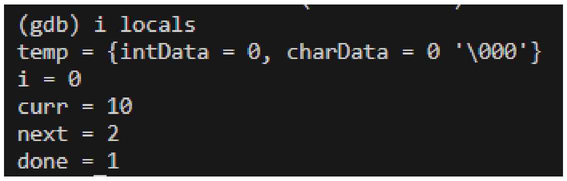
After fixing the logic:



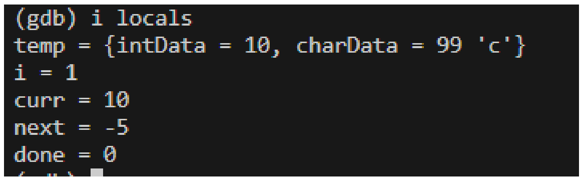
```
temp.intData = array[i].intData;
temp.charData = array[i].charData;

// array[i].intData = temp.intData;
// array[i].charData = temp.charData;
array[i].intData = array[i+1].intData;  //
array[i].charData = array[i+1].charData; //

//array[i + 1].intData = array[i].intData;
//array[i + 1].charData = array[i].charData
array[i+1].intData = temp.intData; //array[
array[i+1].charData = temp.charData; //arra
```



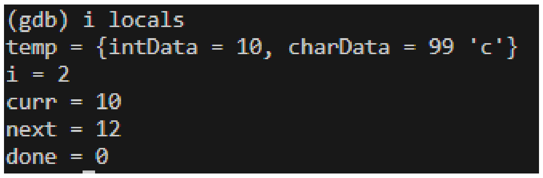
```
(gdb) break Question4.c:29
Breakpoint 1 at 0x404880: file Question4.c, line 29.
(gdb) break Question4.c:35
Breakpoint 2 at 0x4048b7: file Question4.c, line 35.
```



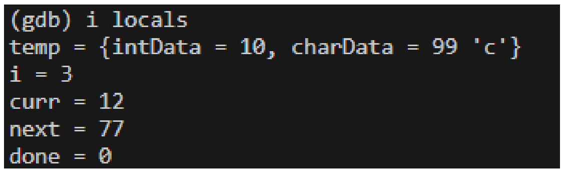
```
(gdb) i locals
temp = {intData = 0, charData = 0 '\000'}
i = 0
curr = 10
next = 2
done = 1
```



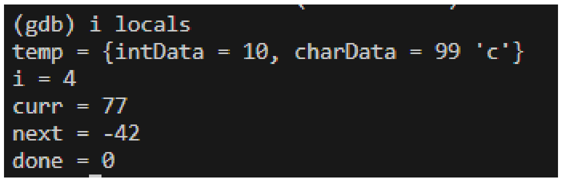
```
(gdb) i locals
temp = {intData = 10, charData = 99 'c'}
i = 1
curr = 10
next = -5
done = 0
```



```
(gdb) i locals
temp = {intData = 10, charData = 99 'c'}
i = 2
curr = 10
next = 12
done = 0
```

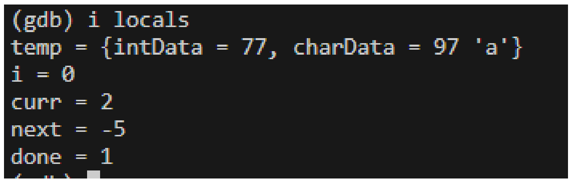


```
(gdb) i locals
temp = {intData = 10, charData = 99 'c'}
i = 3
curr = 12
next = 77
done = 0
```

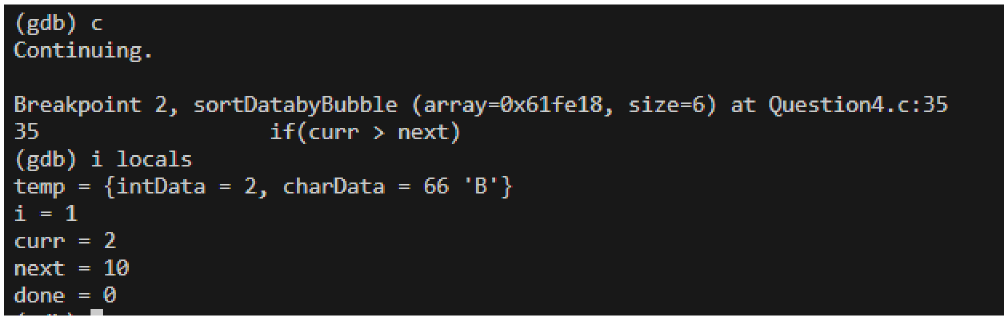


```
(gdb) i locals
temp = {intData = 10, charData = 99 'c'}
i = 4
curr = 77
next = -42
done = 0
```

Enters second iteration of the while loop as sorting was not completed.



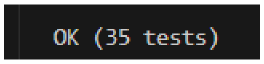
```
(gdb) i locals
temp = {intData = 77, charData = 97 'a'}
i = 0
curr = 2
next = -5
done = 1
```



```
(gdb) c
Continuing.

Breakpoint 2, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) i locals
temp = {intData = 2, charData = 66 'B'}
i = 1
curr = 2
next = 10
done = 0
```

We can see that the values are being assigned correctly and ordered correctly in the second iteration of the while loop.

Passing all the tests:



```
OK (35 tests)
```