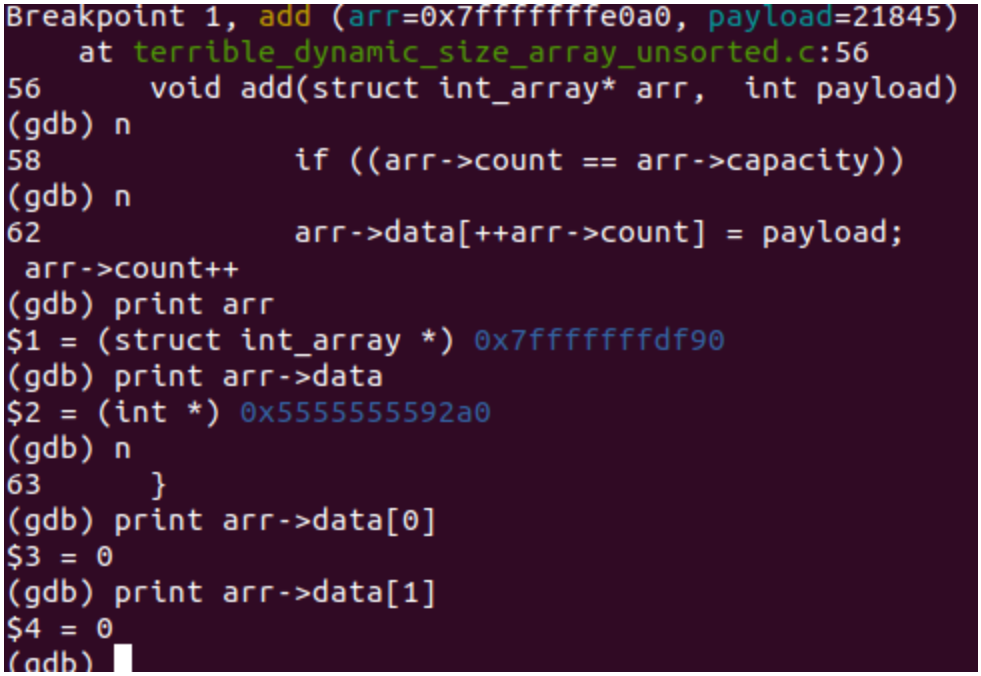
```
(gdb) n
49              arr->data = new_data;
(gdb) n
50              free(arr->data);
(gdb) n
52      }
(gdb) print arr->data[0]
$4 = 0
(gdb) print arr->data[2]
$5 = 1431670800
(gdb) print arr->data[3]
$6 = 21845
```

```
0 0 2 4
0 0 2 4 6
0 0 2 4 6 8
0 0 2 4 6 8 10
0 0 2 4 6 8 10 12
0 0 2 4 6 8 10 12 14
0 0 2 4 6 8 10 12 14 16
0 0 2 4 6 8 10 12 14 16 18
0 0 2 4 6 8 10 12 14 16 18 20
0 0 2 4 6 8 10 12 14 16 18 20 22
0 0 2 4 6 8 10 12 14 16 18 20 22 24
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11
0 0 1431670800 21845 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0
0 0 1431670800 21845 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0 15
0 0 1431670800 21845 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0 15 17
0 0 1431670800 21845 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0 15 17 19
0 0 1431670800 21845 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0 15 17 19 21
```



There were two very obvious printing bugs in this screenshot. One was in resize and the other was in add. Starting with resize, the problem was the use of the free() function. What it did was attempt to free the data and we got left with garbage in indexes 2 and 3. Removing it fixed the issue. The second issue in add would be behind the extra 0's. This was due to it not counting arr->count correctly, because the ++ was before the variable. Since this is an index of an array, we cannot do the ++ before.

```
26      int contains(struct int_array* arr, int target) {
(gdb) n
30              for (i = 0; i < arr->count; ++i);
(gdb) n
32                      if (arr->data[i] == target)
(gdb) n
35                              return FALSE;
(gdb) n
38      }
```

```
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0 15
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0 15 17
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0 15 17 19
0 0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 0 15 17 19 21
Resize function works properly
Number 6 present in Array
```

The next bug was in contains. What would happen is that the if statement had an else condition that would return FALSE as a base case. However, that didnt work because every time the loop was initiated, unless the element we are looking for was the first one, then it would go straight to the else statement and return FALSE. Simply removing it was the fix for this.

```
Number 23 not in Array error in remove_elem
Number 24 not in Array error in remove_elem
Number 0 not in Array error in remove_elem
```

```
65      int remove_elem(struct int_array* arr,  int target) {
(gdb) n
67              unsigned int i = 0;
(gdb) n
69              if ((arr->count = 0))
(gdb) n
76                              return FALSE;
(gdb) n
84      }
(gdb)
```

Lastly came these two bugs in remove_elem. The first one being the if statement comparison. The first if statement did not have a == sign for its comparison, so nothing ended up happening with that one. Skipping 7 lines ahead, we return FALSE, even though its nested in a conditional statement. The reason this broke was due to a semicolon at the end of the statement, preventing it from nesting the return FALSE;. After this fix, everything was working as intended.

```
Starting program: /home/maciej/Desktop/CS382/Labs/Lab_4/a.out
E ay created
Array cleared of data
0
0 2
0 2 4
0 2 4 6
0 2 4 6 8
0 2 4 6 8 10
0 2 4 6 8 10 12
0 2 4 6 8 10 12 14
0 2 4 6 8 10 12 14 16
0 2 4 6 8 10 12 14 16 18
0 2 4 6 8 10 12 14 16 18 20
0 2 4 6 8 10 12 14 16 18 20 22
0 2 4 6 8 10 12 14 16 18 20 22 24
0 2 4 6 8 10 12 14 16 18 20 22 24 1
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 13
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 13 15
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 13 15 17
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 13 15 17 19
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 13 15 17 19 21
0 2 4 6 8 10 12 14 16 18 20 22 24 1 3 5 7 9 11 13 15 17 19 21 23
Resize function works properly
Number 6 present in Array
Number 30 not in Array
Number 23 removed from Array
Number 24 removed from Array
Number 0 removed from Array
Number not in Array
Array destroyed
```