## Error # 1: Number 6 not in Array Contains not working properly

```
Resize function works properly
Number 6 not in Array Contains not working properly
Number 30 not in Array
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
Number not in Array
free(): double free detected in tcache 2

Program received signal SIGABRT, Aborted.
__GI_raise (sig=sig@entry=6) at ../sysdeps/unix/sysv/linux/raise.c:50
50      ../sysdeps/unix/sysv/linux/raise.c: No such file or directory.
```

We can see that the contains function isn't properly working so 6 isn't showing up in the array.

```
Breakpoint 1, contains (arr=0x7fffffffdf10, target=21845)
    at terrible_dynamic_size_array_unsorted.c:26
26      int contains(struct int_array* arr, int target) {
(gdb) step
30              for (i = 0; i < arr->count; ++i);
(gdb) step
32                      if (arr->data[i] == target)
(gdb) step
35                              return FALSE;
(gdb) step
```

I set a breakpoint at line 26 and tried to go through the steps using step. I noticed that the program was clearly running only once and returning false. This means there is a logical error.

```
int contains(struct int_array* arr, int target) {

    unsigned int i;

    for (i = 0; i < arr->count; ++i)
    {
        if (arr->data[i] == target)
            return TRUE;
    }
    return FALSE;
}
```

Based on the debugger, I saw that the loop only checked the first output and returned False. Based on this, I removed the else statement in the for loop and the semicolon in front of the for loop condition.

```
Resize function works properly
Number 6 present in Array
Number 30 not in Array
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
Number not in Array
free(): double free detected in tcache 2

Program received signal SIGABRT, Aborted.
__GI_raise (sig=sig@entry=6) at ../sysdeps/unix/sysv/linux/raise.c:50
50      ../sysdeps/unix/sysv/linux/raise.c: No such file or directory.
```

Once I fixed this function the feedback output showed that the function now works and 6 is in fact present.

## Error #2: Number 23 not in Array error in remove_elem

```
(gdb) break terrible_dynamic_size_array_unsorted.c: 61
```

Since the remove_elem function wasn't working, I set a breakpoint at line 61 where that function is defined. After implement

```
(gdb) step
main () at test_int_array.c:48
48              if (remove_elem(&arraytest, 23) && arraytest.count == 24)
(gdb) step
50              else printf("Number 23 not in Array error in remove_elem\n");
(gdb) step
__GI__IO_puts (str=0x555555556168 "Number 23 not in Array error in remove_elem")
    at ioputs.c:33
33      ioputs.c: No such file or directory.
```

As you can see the error was in the if statement as it should have a || instead of &&.

```
int remove_elem(struct int_array* arr,  int target) {

    unsigned int i = 0;

    if ((arr->count = 0))

        return FALSE;

    while (i < arr->count || arr->data[i] != target)  i++;
```

This is my fixed code.

## Error #3: Number 24 and 0 not in Array error in remove_elem

I first set breakpoint at line 62 and started running through the function by using the next command. As I went through each line I noticed that there were many logical errors and that needed to be fixed for

```
Breakpoint 1, remove_elem (arr=0x7fffffffdf10, target=21845)
    at terrible_dynamic_size_array_unsorted.c:61
61      int remove_elem(struct int_array* arr,  int target) {
(gdb) next
63              unsigned int i = 0;
(gdb) next
65              if ((arr->count = 0))
(gdb) next
74                      return FALSE;
(gdb) next
81      }
(gdb) next
main () at test_int_array.c:48
48              if (remove_elem(&arraytest, 23) && arraytest.count == 24)
```

```c
int remove_elem(struct int_array* arr,  int target) {

    unsigned int i = 0;

    if ((arr->count == 0)){ //Changed = to == and added brackets
        return FALSE;
    }

    while (arr->data[i] != target){ //Replaced the && with a || and moved the first conditional statement to inside of the loop

        if (i >= arr->count){ //changed == to >=
            return FALSE;
            }

        i++;
    }

    arr->data[i] = arr->data[arr->count];

    arr->count--;

    return TRUE;
```

## Error #4: Free(): double free detected in tcache 2

I set a breakpoint at line 38 but after using the continue command, I realized that the error wasn't because of any syntax, but because of a logical error. After setting a break at line 38 and using the continue function, the result was correct and now error was output except for the free() error. So I went deeper into my code and learned that I had to free up the space before using it.

```
Breakpoint 1, resize (arr=0x555555555100 <_start>)
    at terrible_dynamic_size_array_unsorted.c:38
38      void resize(struct int_array* arr) {
(gdb) c
Continuing.
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
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
free(): double free detected in tcache 2
```

```
void resize(struct int_array* arr) {

    arr->capacity *= 2;
    int* new_data = (int*)malloc(sizeof(int) * arr->capacity);
    for (unsigned int i = 0; i < arr->count; ++i)
    {
        new_data[i] = arr->data[i];
    }
    free(arr->data);//Put free before arr->data = new_data;
    arr->data = new_data;

}
```

x

# Error #5: extra row of 0s

**This is a picture of the bottom half of the triangle as you can see the function was outputting an extra row of 0s**

```
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
```

```
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
free(): double free detected in tcache 2
```

**Like error 4, after running the debugger program, there were no errors detected, so I used break points at each different section allowing me to find the error in the add function. Changing the placement of ++ fixed the issue as shown down below.**

```
void add(struct int_array* arr,  int payload) {

    if (arr->count == arr->capacity)

        resize(arr);

    arr->data[arr->count++] = payload; //we changed the placement of ++
}
```

```
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
```

# Completed code:

**After completing my the final bug, I recompiled and ran the program again and finally got the correct output.**