

Name: Harris Spahic

Pledge: “I pledge my honor I have abided by the Stevens Honor System.”

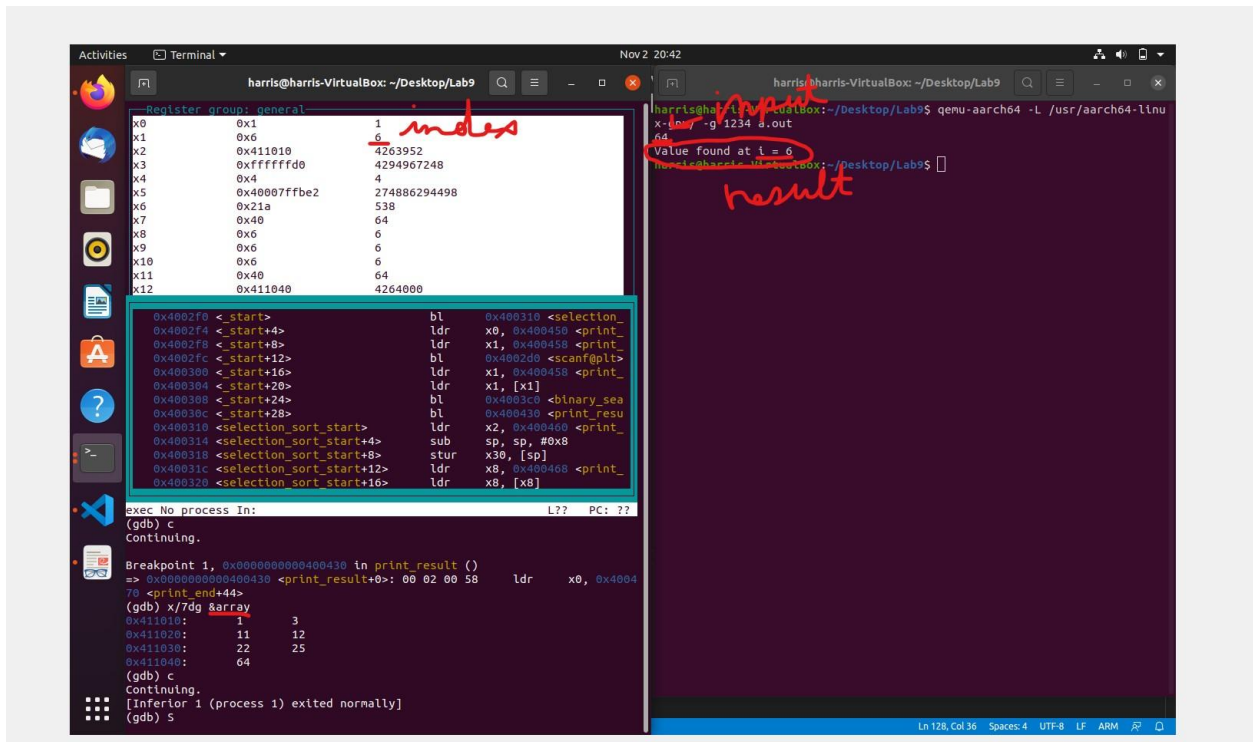
Writeup:

```
harris@harris-VirtualBox: ~/Desktop/Lab9
Register group: general
x0 0x1 1
x1 0x64 100
x2 0x411010 4263952
x3 0xfffff0 4294967248
x4 0x0 0
x5 0x40007ffbe3 274886294499
x6 0x21a 538
x7 0x64 100
x8 0x6 6
x9 0x7 7
x10 0x6 6

0x4002f0 <_start> bl 0x400310 <selection
0x4002f4 <_start+4> ldr x0, 0x400450 <print
0x4002f8 <_start+8> ldr x1, 0x400450 <print
0x4002fc <_start+12> bl 0x4002d0 <scanf@plt
0x400300 <_start+16> ldr x1, 0x400458 <print
0x400304 <_start+20> ldr x1, [x1]
0x400308 <_start+24> bl 0x4003c0 <binary_se
0x40030c <_start+28> bl 0x400430 <print_res
0x400310 <selection_sort_start> ldr x2, 0x400460 <print
0x400314 <selection_sort_start+4> sub sp, sp, #0x8
0x400318 <selection_sort_start+8> stur x30, [sp]
0x40031c <selection_sort_start+12> ldr x8, 0x400460 <print

exec: No process in: L77 PC: ??
(gdb) x/7dg $array
0x411010: 1 3
0x411020: 11 12
0x411030: 22 25
0x411040: 64
(gdb) c
Continuing.
[Inferior 1 (process 1) exited normally]
(gdb) s
```

Case 1: In this case, I give an input that is not in my array (100). My program runs through binary search, has the left pointer edge surpass the right edge of the array and branches off to the “print_error” branch since the value was not found. Hence, x1 does not change to the index of (100) in my array since it was never found.



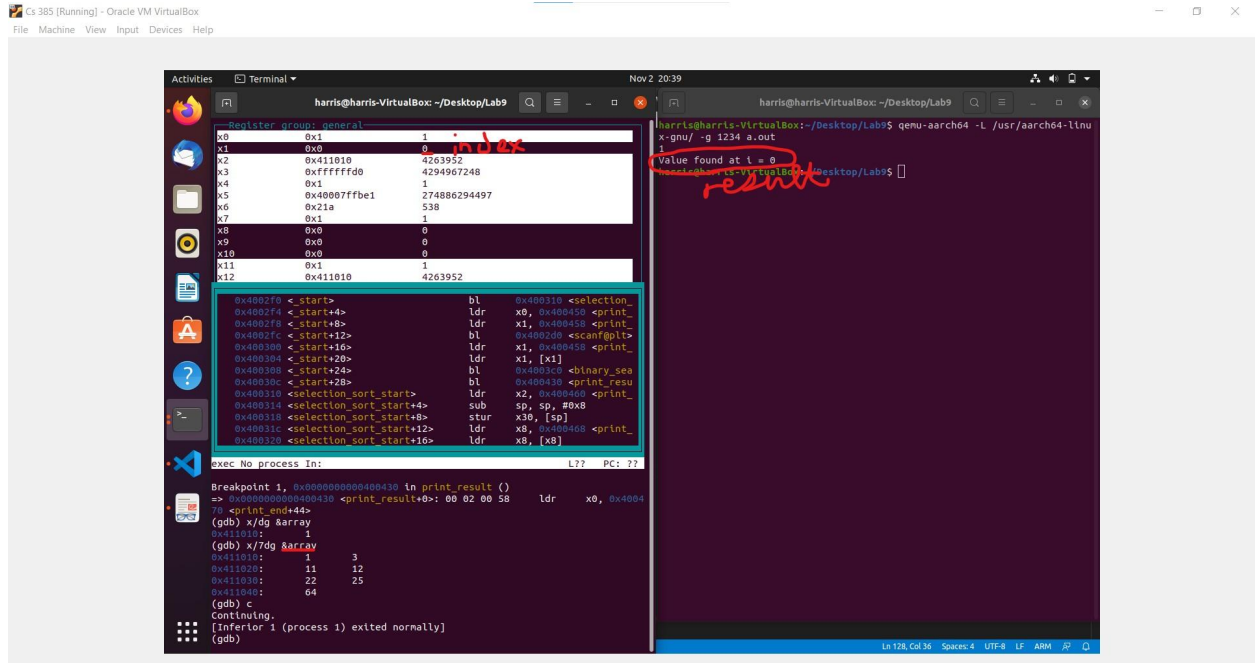
Case 2: Input is the last element of my sorted array. In this case binary_search repeatedly takes the right half of the array until it finds the first element. The index of the said element is stored into x1. Then binary_search branches to print_result, which prints that the “value was found at i = 6”. Since my size is 7, that is correct.

```
harris@harris-VirtualBox: ~/Desktop/Lab9
Register group: General
x0 0x1 1
x1 0x4 4
x2 0x411010 4263952
x3 0xffffffff 4294967248
x4 0x2 2
x5 0x40007ffbe2 274886294498
x6 0x21a 538
x7 0x16 22
x8 0x4 4
x9 0x4 4
x10 0x4 4
x11 0x16 22
x12 0x411030 4263984

0x4002f0 <_start> b1 0x400210 <selection_
0x4002f4 <_start+4> ldr x0, 0x400450 <print_
0x4002f8 <_start+8> ldr x1, 0x400458 <print_
0x4002fc <_start+12> bl 0x400200 <scanf@plt>
0x400300 <_start+16> ldr x1, 0x400450 <print_
0x400304 <_start+20> ldr x1, [x1]
0x400308 <_start+24> bl 0x4003c0 <binary_sea
0x40030c <_start+28> bl 0x400430 <print_resu
0x400310 <selection_sort_start> ldr x2, 0x400450 <print_
0x400314 <selection_sort_start+4> sub sp, sp, #0x8
0x400318 <selection_sort_start+8> stur x30, [sp]
0x40031c <selection_sort_start+12> ldr x8, 0x400460 <print_
0x400320 <selection_sort_start+16> ldr x8, [x8]

exec no process in: L77 PC: 77
(gdb) c
Continuing.
Breakpoint 1, 0x400430 in print_result ()
> 0x400430: 0x400430 <print_result+0>: 00 02 00 58 ldr x0, 0x4004
70 <print_end+44>
(gdb) x/7dg $array
0x411010: 1 3
0x411020: 11 12
0x411030: 22 25
0x411040: 64
(gdb) c
Continuing.
[Inferior 1 (process 1) exited normally]
(gdb)
```

Case 3: Input is the middle element of my sorted array. I chose 22, the 5th element in my array. In this case binary_search alternates accordingly for splitting the left and right half of the array. The index of the said element is stored into x1. Then binary_search branches to print_result, which prints that the “value was found at i = 4”. Since i = 0 is the first element this is correct.



Case 3: Input is the first element of my sorted array. In this case `binary_search` takes the right half of the array repeatedly until it reaches the end. The index of the said element is stored into `x1`. Then `binary_search` branches to `print_result`, which prints that the “value was found at `i = 0`”. Since `i = 0` is the first element this is correct.