CS-35101

Thomas Moore

3/6/22

Summary:

For lab2 I implemented a single if else statement on lines 52 and 47 using the branch greater than or equal function. This allowed me to only have to use subtraction on line 44 and store the value to compare to w on line 45. Problem a on lab2 was able to be done with a simple swapping idiom between lines 29 and 31. Problem b took a bit more thinking and required arithmetic to swap without a third variable on lines 29 through 31 of lab2-b.asm. The core of the code of lab2-a and lab2-b are the same as lab2.asm as the only things that really needed to be changed were the inputs for the values of the registers and to add a swapping mechanism.

Conclusion:

The main problem I faced was in problem b where it required a swap to be done with no other registers available. It took some time and help from a tutor to figure out a way to mathematically do the same thing as swapping the variables, but it is indeed possible. This has taught me a valuable lesson for the future when I am coding in assembly due to the limited number of registers available. If I ever run into a situation where I need to swap register values with no available registers I am confident that I will be able to do so now.

lab2.asm

```
1.  #Thomas Moore
2.  .data
3.          w: .asciiz "Enter value for w: "
4.          x: .asciiz "Enter value for x: "
5.          y: .asciiz "Enter value for y: "
6.          z: .asciiz "Enter value for z: "
7.          printx: .asciiz "The value of x is: "
8.
9.  .text
10.
11. #input w
12.         li $v0, 4
13.         la $a0, w
14.         syscall
15.         li $v0, 5
```

```
16.             syscall
17.             add $t1, $0, $v0 #$t1 = w
18.
19.     #input x
20.             li $v0, 4
21.             la $a0, x
22.             syscall
23.             li $v0, 5
24.             syscall
25.             add $t2, $0, $v0 #$t2 = x
26.
27.     #input y
28.             li $v0, 4
29.             la $a0, y
30.             syscall
31.             li $v0, 5
32.             syscall
33.             add $t3, $0, $v0 #$t3 = y
34.
35.     #input z
36.             li $v0, 4
37.             la $a0, z
38.             syscall
39.             li $v0, 5
40.             syscall
41.             add $t4, $0, $v0 #$t4 = z
42.
43.     #subtraction and if statment
44.             sub $t5, $t2, $t3 # x-y = $t5
45.             bge $t5, $t1, if # if [(x-y)>= w] go to if:
46.
47.     else:
48.             #set x to z
49.             move $t2, $t4
50.             j print
51.
52.     if:
53.             #set x to y
54.             move $t2, $t3
55.             j print
56.
57.     print:
58.             li $v0, 4
59.             la $a0, printx
```

```
60.             syscall
61.             move $a0, $t2 #moving x to $a0
62.             li $v0, 1
63.             syscall
64.
65.     terminate:
66.             li $v0, 10
67.             Syscall
```

lab2-a.asm

```
1.  #Thomas Moore
2.  .data
3.              r1: .asciiz "Enter value for register1: "
4.              r2: .asciiz "Enter value for register2: "
5.
6.              printswap: .asciiz "\nThe values have been swapped "
7.              printr1: .asciiz "\nThe value stored in r1 is: "
8.              printr2: .asciiz "\nThe value stored in r2 is: "
9.
10. .text
11.
12. #input r1
13.             li $v0, 4
14.             la $a0, r1
15.             syscall
16.             li $v0, 5
17.             syscall
18.             add $s1, $0, $v0 #$s1 = r1
19.
20. #input r2
21.             li $v0, 4
22.             la $a0, r2
23.             syscall
24.             li $v0, 5
25.             syscall
26.             add $s2, $0, $v0 #$s2 = r2
27.
28. #swap r1 and r2
29.             move $t0, $s2 #set $t0 to value of r2
30.             move $s2, $s1 #set r2 to r1
31.             move $s1, $t0 #set r1 to value of r2 via temp register
32. print:
33.             li $v0, 4
```

```
34.         la $a0, printswap
35.         syscall
36.
37.         li $v0, 4
38.         la $a0, printr1
39.         syscall
40.         move $a0, $s1 #moving r1 to $a0
41.         li $v0, 1 #print int
42.         syscall
43.         li $v0, 4
44.         la $a0, printr2
45.         syscall
46.         move $a0, $s2 #moving r2 to $a0
47.         li $v0, 1
48.         syscall
49.
50. terminate:
51.         li $v0, 10
52.         Syscall
```

lab2-b.asm

```
1.  #Thomas Moore
2.  .data
3.          r1: .asciiz "Enter value for register1: "
4.          r2: .asciiz "Enter value for register2: "
5.
6.          printswap: .asciiz "\nThe values have been swapped "
7.          printr1: .asciiz "\nThe value stored in r1 is: "
8.          printr2: .asciiz "\nThe value stored in r2 is: "
9.
10. .text
11.
12. #input r1
13.         li $v0, 4
14.         la $a0, r1
15.         syscall
16.         li $v0, 5
17.         syscall
18.         add $s1, $0, $v0 #$s1 = r1
19.
20. #input r2
21.         li $v0, 4
22.         la $a0, r2
```

```
23.         syscall
24.         li $v0, 5
25.         syscall
26.         add $s2, $0, $v0 #$s2 = r2
27.
28. #swap via mathmatics
29.         add $s1, $s1, $s2 # r1 = r1 + r2
30.         sub $s2, $s1, $s2 # r2 = r1 - r2
31.         sub $s1, $s1, $s2 # r1 = r1 - r2
32.
33. print:
34.         li $v0, 4
35.         la $a0, printswap
36.         syscall
37.
38.         li $v0, 4
39.         la $a0, printr1
40.         syscall
41.         move $a0, $s1 #moving r1 to $a0
42.         li $v0, 1 #print int
43.         syscall
44.         li $v0, 4
45.         la $a0, printr2
46.         syscall
47.         move $a0, $s2 #moving r2 to $a0
48.         li $v0, 1
49.         syscall
50.
51. terminate:
52.         li $v0, 10
53.         Syscall
```

Results:

lab2.asm

```
Enter value for w: 5
Enter value for x: 6
Enter value for y: 3
Enter value for z: 1
The value of x is: 1
-- program is finished running --

Enter value for w: 10
Enter value for x: 15
Enter value for y: 3
Enter value for z: 1
The value of x is: 3
-- program is finished running --
```

lab2-a.asm

```
Enter value for register1: 7
Enter value for register2: 5

The values have been swapped
The value stored in r1 is: 5
The value stored in r2 is: 7
-- program is finished running --
```

lab2-b.asm

```
Enter value for register1: 6
Enter value for register2: 9

The values have been swapped
The value stored in r1 is: 9
The value stored in r2 is: 6
-- program is finished running --
```