Matthew Michael Sherlin
Dr. Augustine Samba
Computer Organization
October 29, 2020

**Assembly Code File part 1:**

---

```
.data
	prompt1: .asciiz "If Else statement for [(x-y) >= w] with user input.\nIf (x-y) is greater or equal to w, x will be set to y.\nOtherwise, x will be set to z.\n"
	prompt2: .asciiz "\nEnter a value for w: "
	prompt3: .asciiz "Enter a value for x: "
	prompt4: .asciiz "Enter a value for y: "
	prompt5: .asciiz "Enter a value for z: "
	message1: .asciiz "\nSetting x to y."
	message2: .asciiz "\nSetting x to z. "
	message3: .asciiz "\n\nThe value of x is now: "

.text

li $v0, 4		#reading the first prompt
la $a0, prompt1
syscall

li $v0, 4		#reading the second prompt
la $a0, prompt2
syscall
li $v0, 5		#entering user input
syscall
move $s0, $v0		#moving user input into saved register

li $v0, 4		#reading the third prompt
la $a0, prompt3
syscall
li $v0, 5		#entering user input
syscall
move $s1, $v0		#moving user input into saved register

li $v0, 4		#reading the fourth prompt
la $a0, prompt4
syscall
li $v0, 5		#entering user input
syscall
move $s2, $v0		#moving user input into saved register

li $v0, 4		#reading the fifth prompt
la $a0, prompt5
syscall
```

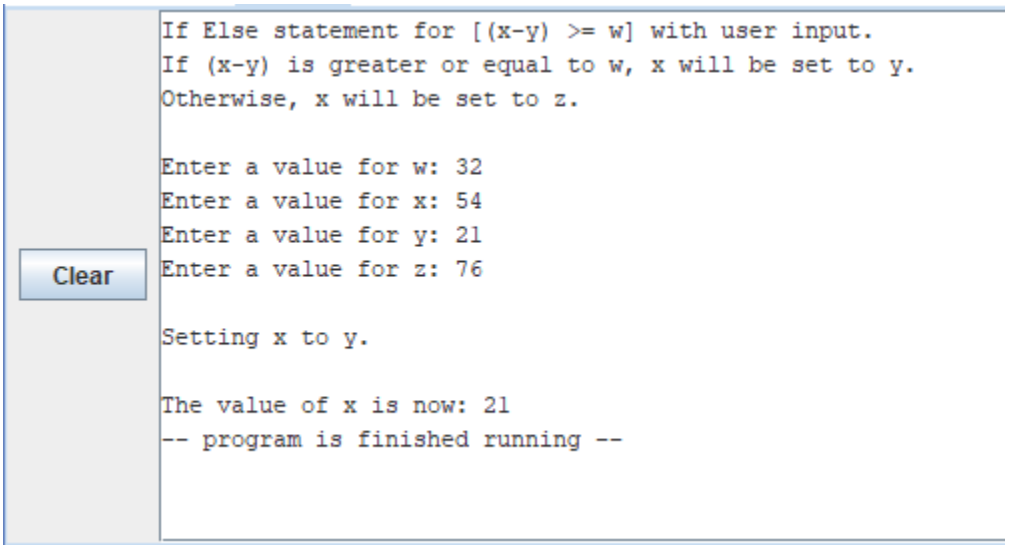
```
li $v0, 5            #entering user input
syscall
move $s3, $v0            #moving user input into saved register

sub $t4, $s1, $s2        #subtracting
blt $t4, $s0, else       #test if x-y < if so go to else:
li $v0, 4            #read prompt
la $a0, message1
syscall
move $s1, $s2            #store value of y into x
j endif                  #jump to endif
else:
li $v0, 4            #read prompt
la $a0, message2
syscall
move $s1, $s3            #store value of z into x
endif:

li $v0, 4            #reading the final message
la $a0, message3
syscall

li $v0, 1            #displaying final answer
move $a0, $s1
syscall

li $v0, 10           #terminate program run and exit
syscall
```

---

**Project Implementation:**

To begin this project, I used basic li and la instructions to read the prompts and retrieve the inputs from the console. As per Dr. Samba, I was allowed to use 'move' instruction for the first section of this lab, so I used 'move' to move the values entered into saved registers. Next, in order to see if w was less than or equal to, I used sub first to find x-y. Next, I used 'blt' or branch less than instruction to see if w was less than x-y. I found this simpler than doing and equals as well as a greater than instruction. If it were found to be less than, I created an else statement that would then do the swapping of the variables using move instruction. Otherwise, I did the swapping the same way and added a jump to the end of the if statement. Finally, I displayed the final answer using basic syscall statements and terminated the program.

**Working Code Screen Print:**




Transcription:

If Else statement for [(x-y) >= w] with user input.
If (x-y) is greater or equal to w, x will be set to y.
Otherwise, x will be set to z.

Enter a value for w: 32
Enter a value for x: 54
Enter a value for y: 21
Enter a value for z: 76

Setting x to y.

The value of x is now: 21
-- program is finished running --

**Conclusion:**

To conclude, I enjoyed this lab using MIPS also. I learned a couple of new instructions, those being blt (branch less than), bgt (branch greater than) and ble (branch less than equal to). Even though I did not end up using those instructions, through my research while doing this lab I came across these and studied up on them. In terms of issues faced while doing this lab, I came across a problem getting my values to swap in the beginning, but soon after I figured out the issue and got it all sorted out. All in all a great lab.