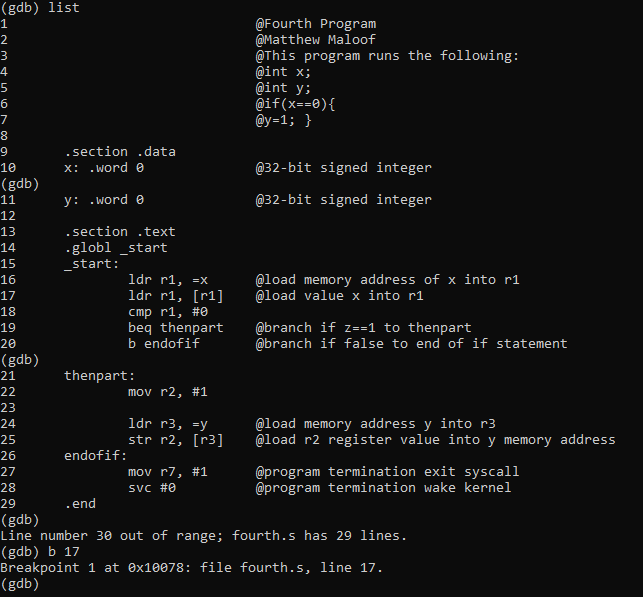
Matthew Maloof

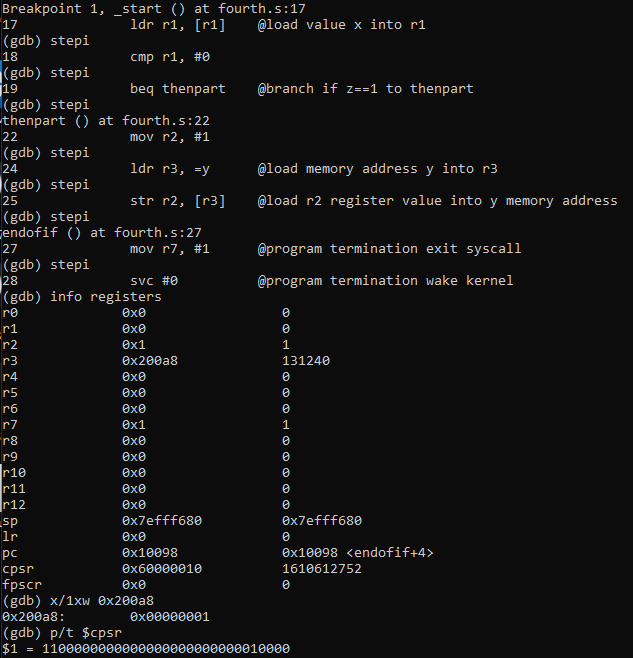
Project 4 Task 4

**Part 1:**



**Figure 1**

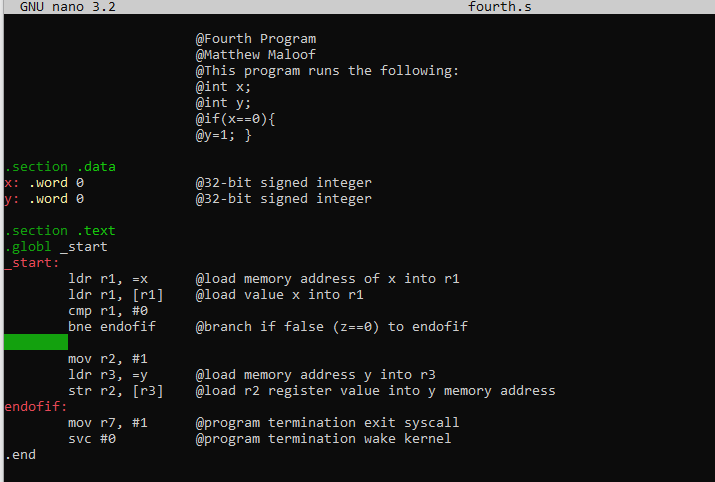
First, I created the file fourth.s using nano command, then copied and pasted the code given in the slides. Then I assembled and linked the file shown above then entered gdb mode.



**Figure 2**

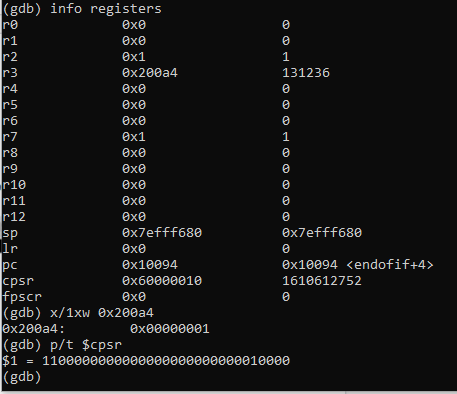
I then set a breakpoint at line 17 which is the start, and used stepi to go through the program. At the end I typed info registers and it displayed the contents of each register, and I used x/1xw command along with my register which is 0x200a8 and it displayed the contents of the 0x200a8 register address which should be 1 because that is what y is set to because x = 0. I then used the p/t $cpsr command to view the flags and it shows the above output. The value 1100000000000000000000000010000 as shown above is equivalent to 32-bits. The bits from right to left show the 29th and 30th as being set, which are the carry flag and zero flag. They are set because subtracting 0 from 0 raises these two flags, showing they are equal numbers.

**Part2:**



**Figure 4**

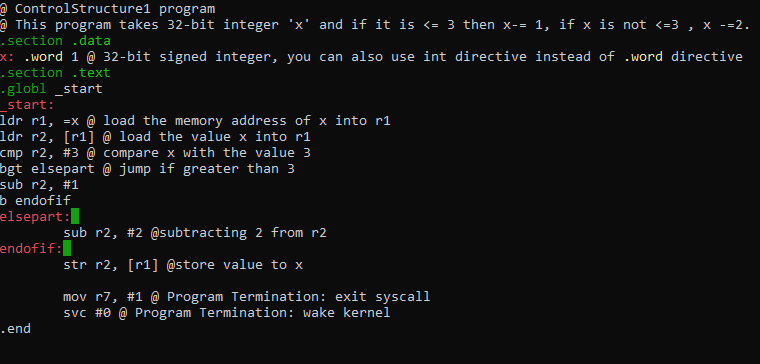
I edited fourth.s by removing the b instruction from the code and replacing beq with bne (branch not equal to zero). Then I assembled and linked the file as shown above. This method allows the program to run more efficiently because there are not back to back branches as in part 1 (beq then b).



**Figure 5**

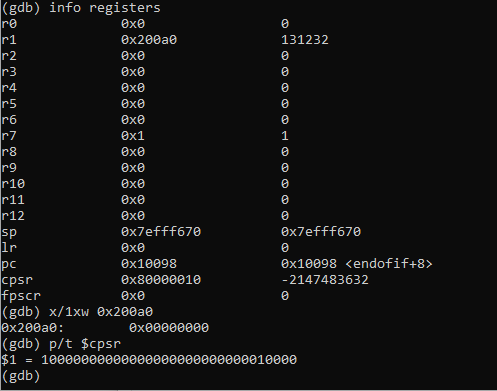
Then I entered debug mode using gdb fourth, and I set a breakpoint at line 17 and used stepi to go through the code. I then used x/1xw to view the contents of 0x200a4 which is my register and the output of 0x00000000 is above, along with p/t $cpsr showing the flags above yielding the same as part 1, zero flag being enabled as the bit is shown as 1. This is a more efficient way to do Part 1’s code.

**Part 3:**



**Figure 6**

I created a new file called ‘ControlStructure1.s’ to calculate the expression if (x <= 3) { x=x-1 } else { x = x - 2}. The code for it is shown above, and I assembled and linked the file then entered gdb debug mode.



**Figure 7**

I set a breakpoint at line 8 and ran the program. I used stepi to go line by line then used p/t $cpsr to view the flag statuses, the negative flag (31st bit) is set because 1 is less than 3 so 3-1 being executed is a negative value.. Typing x/1xw 0x200a0 (address of register 3 which contains the value of x) (Figure 8) shows the contents as 0x00000000 which is correct as x is 1 initially which is <= 3, so it runs x-= 1 which 1-1 = 0.