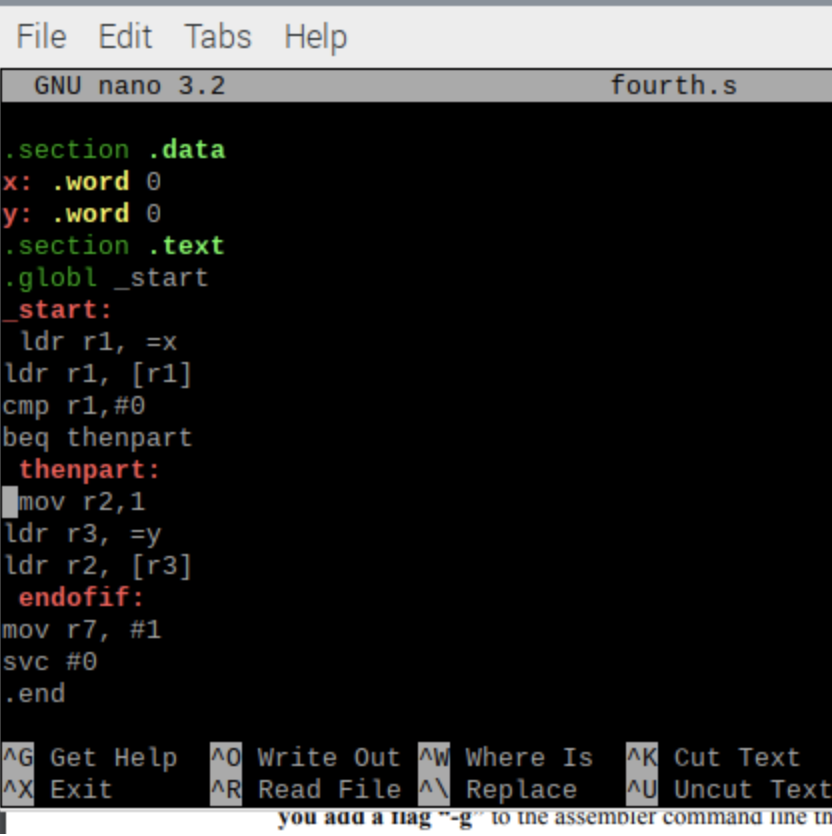
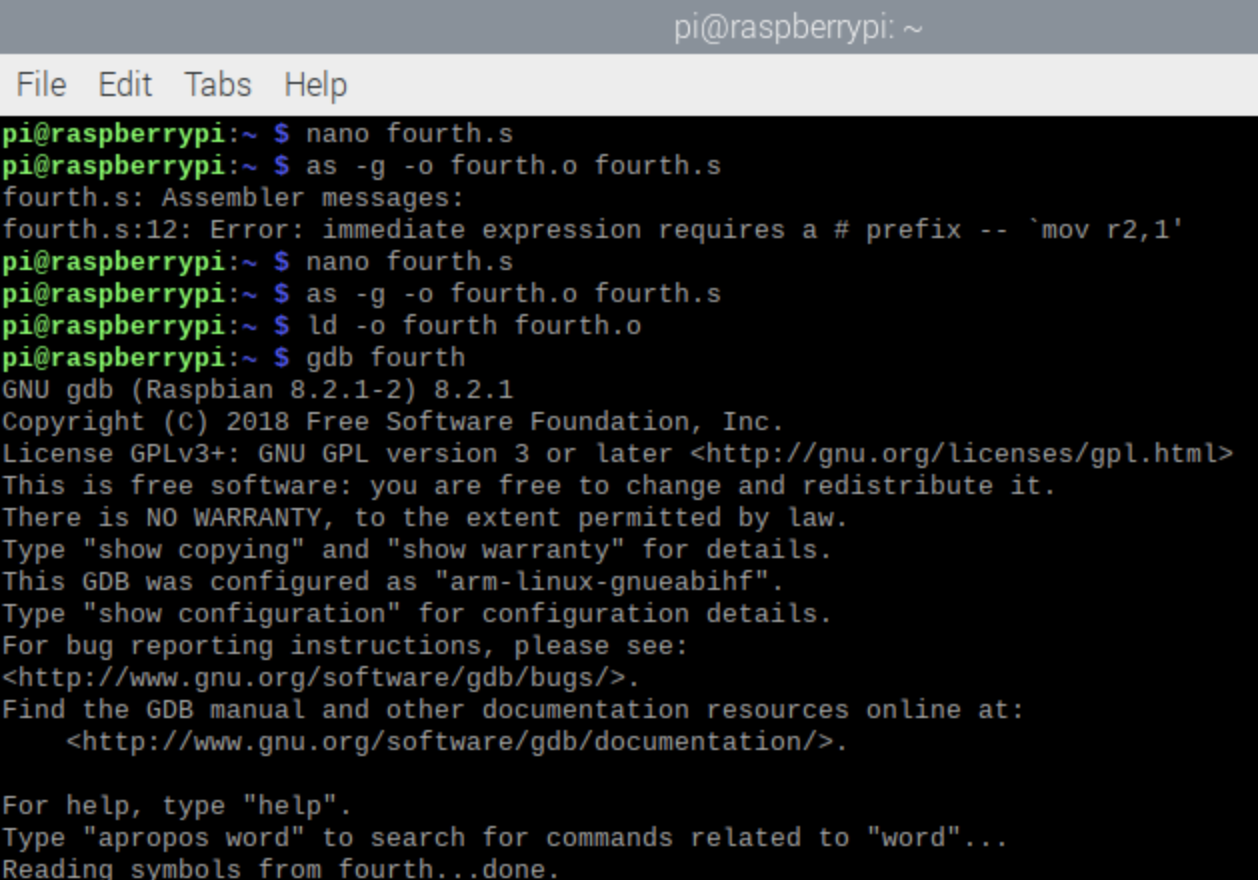
VEDANSI PARSANA

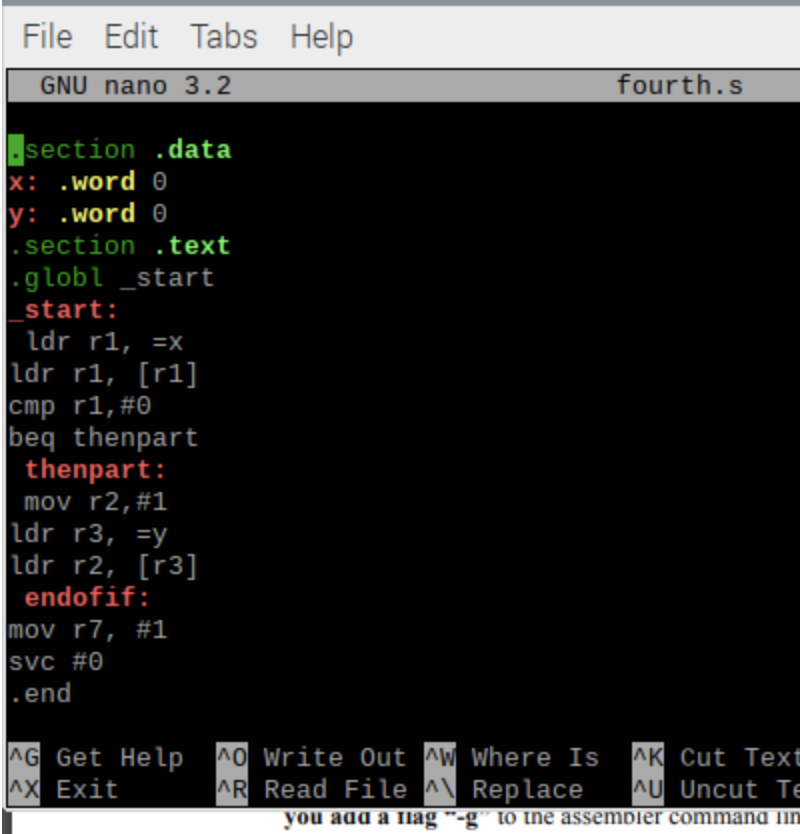
PART 1)

For the fourth program, I read it assembled, linked and executed the file.

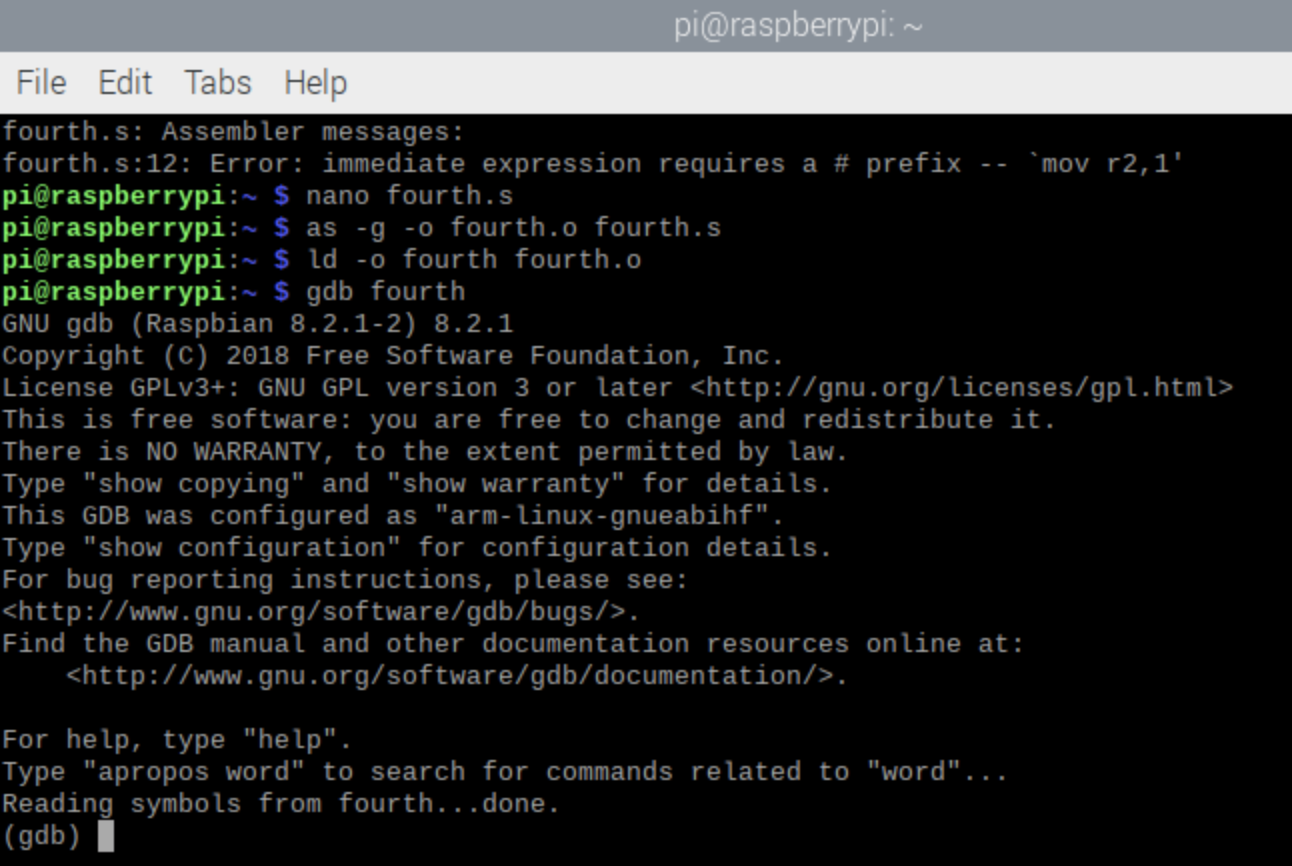


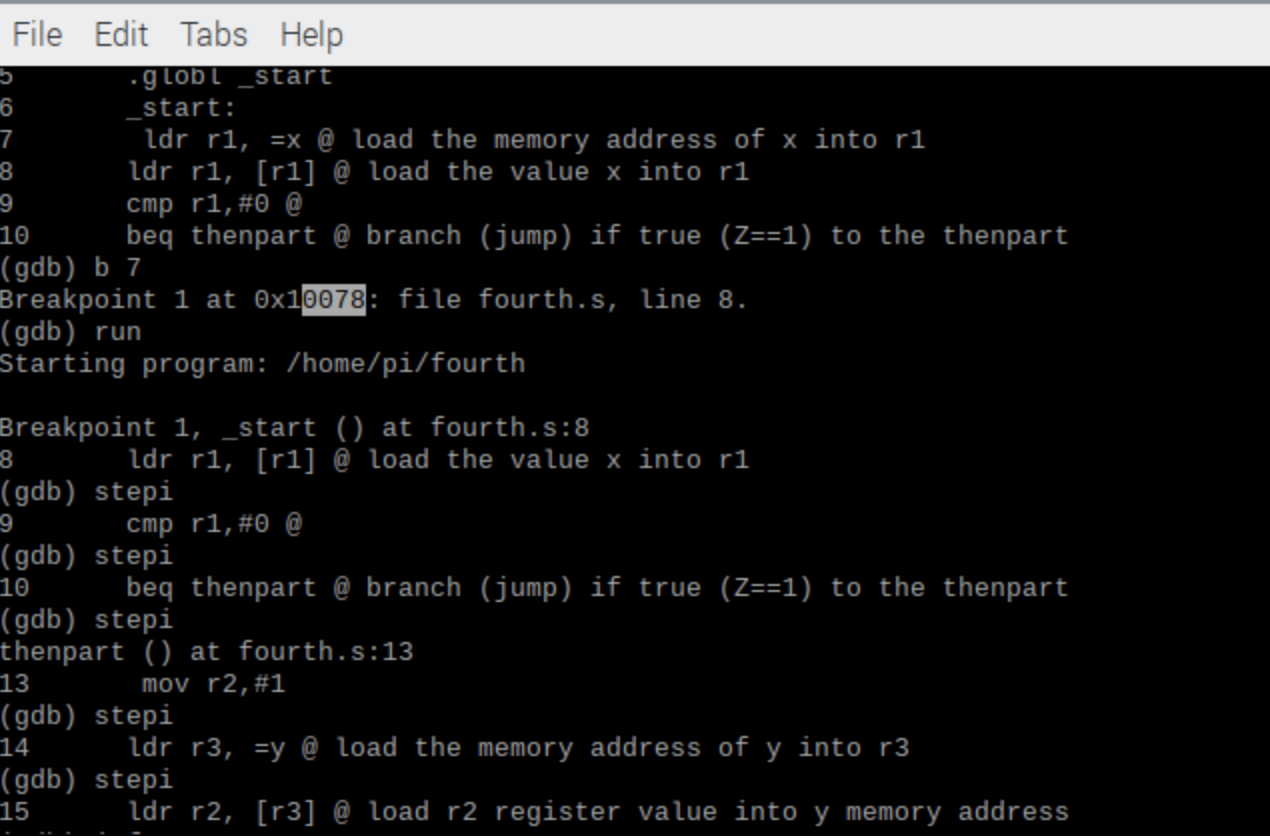
The given code requires a change as the given command ‘mov r2, 1’ is missing a prefix.



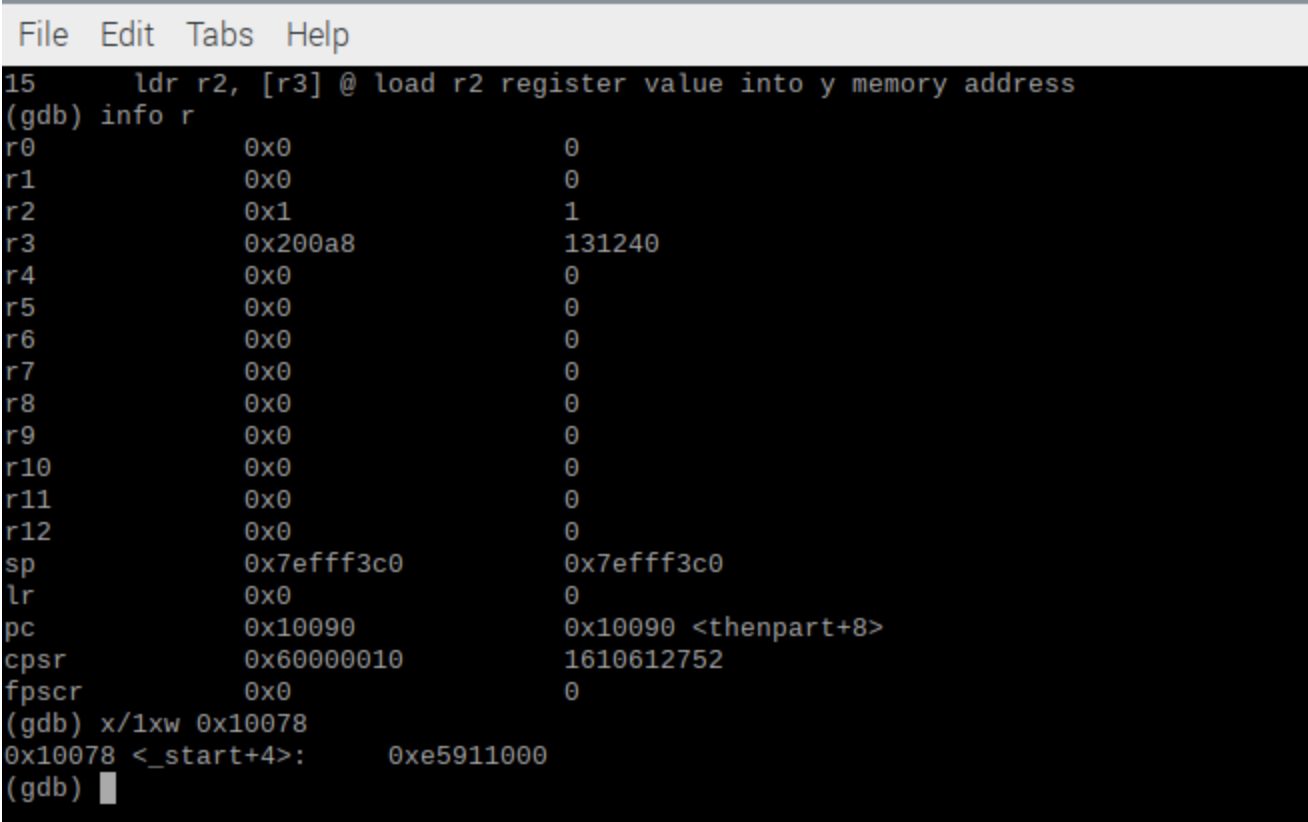


After launching the GNU Debugger, I see the code line numbers using ‘list’ and create a breakpoint at line 7 using ‘b 7’ and test run with ‘run’





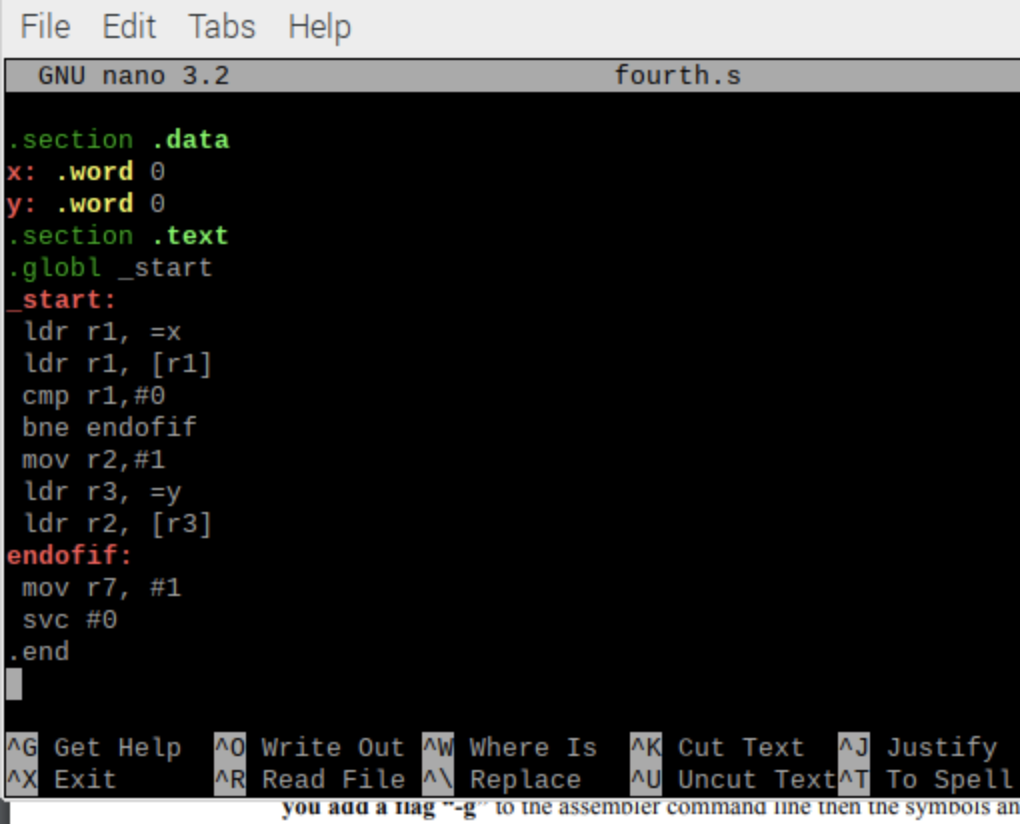
Using ‘stepi’ I continue till before the end, or else it won’t show the registers of the program, then I checked the values of the y memory and the z flag by using ‘x/1xw’ and the given number of our starting point 0x10078, the hexadecimal form 0xe5911000 is where the data is being stored. The register r3, which is holding the y’s address is 0x200a8.



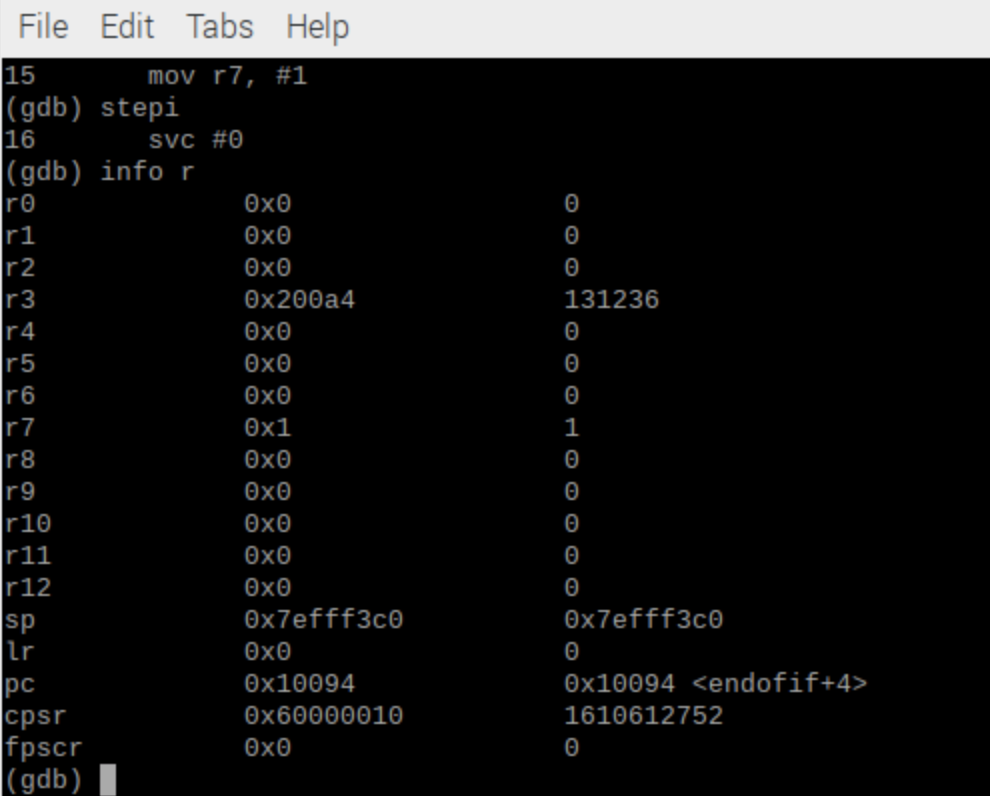
Looking at the registers, we can see that the CPSR is 60000010, meaning the value of the 4 flags (first 4 bits) are 0 1 1 0 to show the flags negative, zero, carry, and 2’s complement OV, respectively. Then the zero flag is set as the second bit is one.

PART 2)

For the second part of this task, I updated the fourth.s code so that it only has one branch or jump. Removing the b jump. Also, the bne jump must be set to now jump to the endofif as it is now the opposite condition as beq.



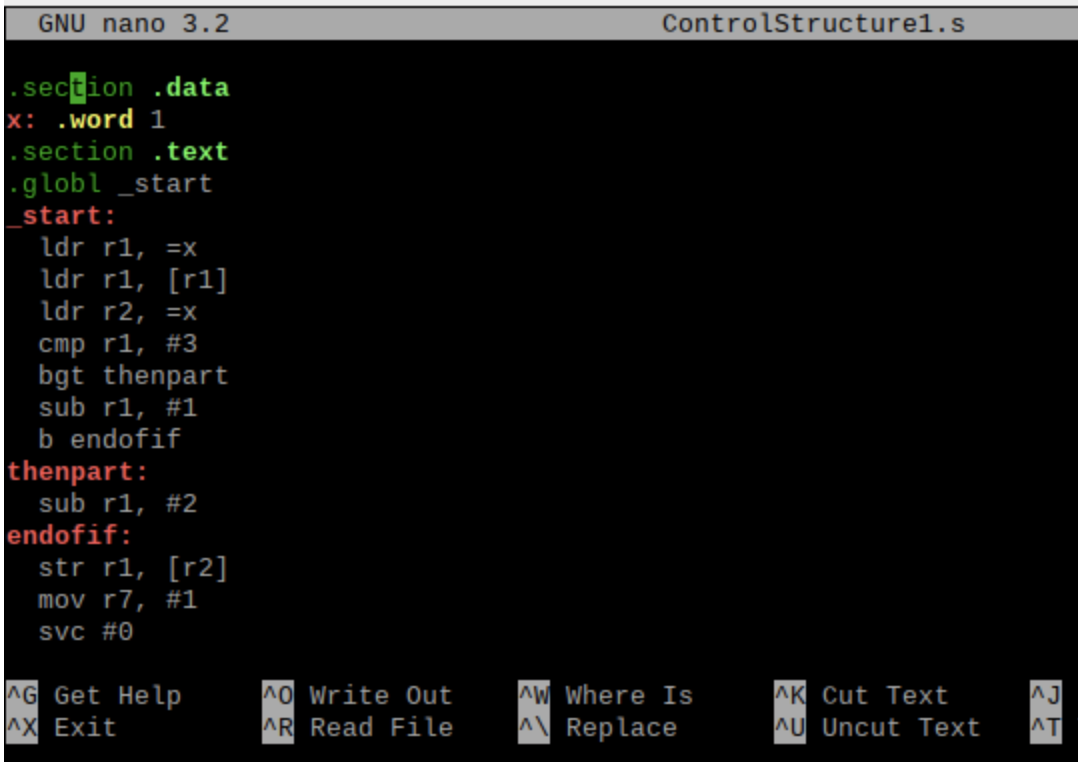
Then I re-assembled, linked and ran it.

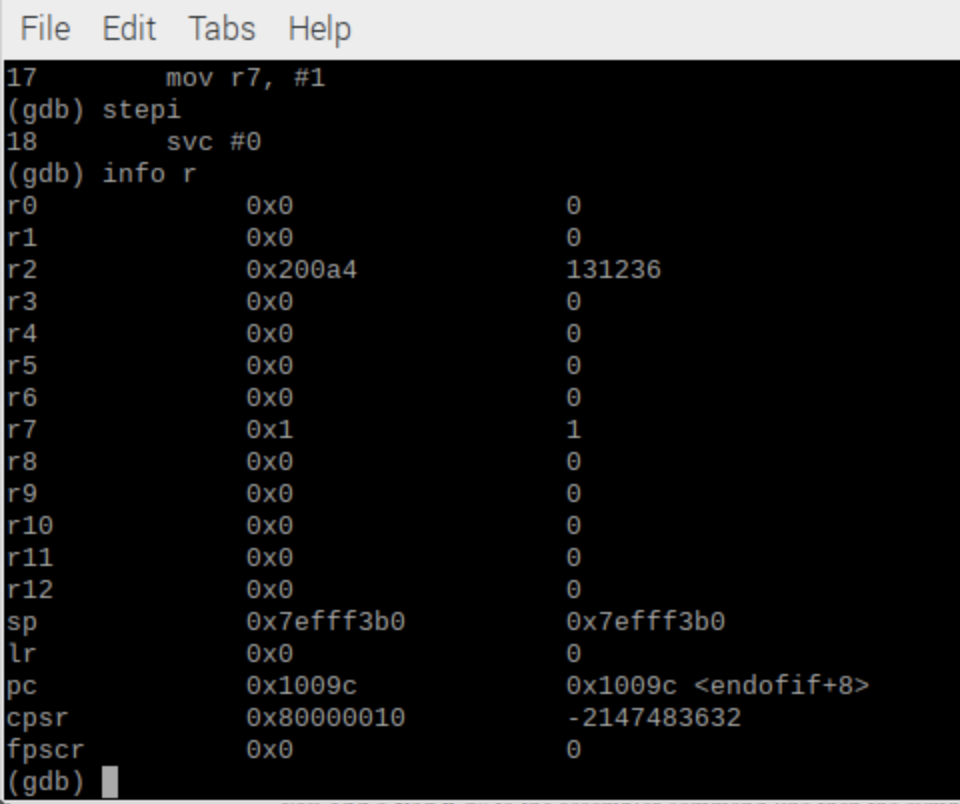


the zero flag is still 1 as the CSPR value as the last 4 bits are 0 1 1 0, since x = 0 compares to 0.

PART 3)

Using the fourth program code again, I wrote the given code for the project.





Using the opposite rule, making the condition x > 3, then subtract 2, otherwise continue past to subtract one and unconditional jump to the end. The value 1 of x is set to 0 at the end, with its address in r2 showing as 0x200a4. The z flag is not set as CPSR ends in bits 1 0 0 0. Since x != 3 then the compare wouldn’t set off the zero flag.