**NextLine (not actually pseudo code)**

#Procedure for finding the next endline character

#expects $a0 to be a location of somewhere in memory, intended for text obviously

#if $a0 points to an endline character then that location will be returned

#returns $v0 the location of the endline character "\n"

nextLine:

#put the scanner location (in a0) in t0

add $t0, $a0, $zero

nextLine\_check:

#get the character pointed to by t0, put in t1

lbu $t1, 0($t0)

#if said character is an endline (\n) go to exit procedure

beq $t1, 13, nextLine\_exit #13 is the decimal value of the carriage return

#otherwise increment t0 and go check again

addi $t0, $t0, 1

j nextLine\_check

#put the location of the \n character in v0

nextLine\_exit:

add $v0, $t0, $zero

jr $ra

**nextNonSpaceChar**

#nextNonSpaceChar

#basically the same thing as nextLine but we want the location in memory to not point to a space

**isComment**

#isComment

#check to see if the argument given is a #

**labelsMatch (this is the fun one)**

#labelsMatch

#pass the location of the last character of a label in $a0

#pass the location of the last character of another label in $a1

#a0, a1 unchanged

#return 1 in v0 if the labels match, return 0 otherwise

#put the locations pointed to by a0 and a1 in t0 and t1 respectively

#check procedure

#load bytes pointed to by t0 and t1 into t2 and t3

#if t2 or t3 is a space ( ) go to final check

#if t2 != t3 go to negative exit

#decrement t0 and t1

#check again

#final check

#if t2 == t3 then both spaces and both labels terminated successfully

#go to positive exit

#negative exit

#put 0 in v0

#exit

#positive exit

#put 1 in v0

#return