# **P4 Test Files - Code Generation**

Compile the input source into the asm and then execute through the virtual machine to assess runtime semantics as noted. The examples are the same for local and global option.

There should be no syntax nor semantics errors so let me know if you see something suspicious.

#p4\_1 - testing output, print 1

main   
 print 1 .  
end

$p4\_2 - testing also input, echo input

main   
 print 1 .  
 scan aa .  
 start  
 let aa = 2 .  
 print aa .  
 stop  
end

$p4\_3 - testing expression, print 1

main   
 print 3 \* 7 / 8 + 5 - - - 4 .   
end

#p4\_4 - testing cond, print 1 if input >0

let aa = 1 .  
main   
 scan aa .  
 cond ( aa > 0 )   
 print 1 .  
end

#p4\_5 - nested cond, print 1 if input is teen age

let aa = 1 .  
main   
 scan aa .  
 cond ( aa >= 13 )   
 cond ( aa <= 19 )  
 print 1 .  
end

$p4\_6 - loop, print input, input-1...down to 1

let aa = 1 .  
main   
 scan aa .   
 loop ( aa > 0 )   
 start  
 print aa .  
 aa ~ aa - 1 .  
 stop  
end

#p4\_7 - testing double loop, print input, input-1...down to 1, then start again with input-2, input-3...down to 1 etc

let aa = 1   
 ab = 2 .  
main   
 scan aa .   
 loop ( aa > 0 )   
 start  
 ab ~ aa .  
 loop ( ab > 0 )  
 start  
 print ab .  
 ab ~ ab - 1 .  
 stop  
 aa ~ aa - 2 .  
 stop  
end