**What to do when you get the error:**

**"Instruction references label that cannot be represented in a 9-bit signed PC offset"**

The problem is that the PC-Relative memory addressing mode (used in LD, LDI, LEA, ST, STI, BR) uses 9 bits to generate the two's complement offset (i.e. the number of memory locations separating the instruction from the label it references).

This means it can only refer to labels that are less than 256 lines before or after the instruction.  
*(JSR also uses PC-Relative addressing mode, but it uses 11 bits, so a subroutine label can be up to 1024 lines before or after the SR call)*.

So what can you do if you get this error?

Sometimes just re-arranging the code and the data can bring all labels back into range - e.g. if you have a large array allocated with .BLKW, just move that to the end.  
But when that fails the only solution is to move the strings to a separate location in memory and provide that location "manually" via a .FILL

The simpl asembler/emulator allows you to to accomplish this using multiple .ORIG entries in a single source file.

**An example:**

.ORIG x3000

; some code

LEA R0, message1 ; load the starting address of message1 into R0.

PUTS ; print the message starting at address message1

; some more code

HALT

; insanely long data block, occupying hundreds of memory locations:

...

message1 .STRINGZ "this is a message"

; it turns out that the address message1 is more than 256 locations distant from the  
; LEA instruction that references it - so we get the dreaded "cannot assemble" error

.END

So what can we do?  
Either move message1 to the start of the data block, or refactor the code like this:

.ORIG x3000

; some code

LD R0, message1\_ptr ; message1\_ptr now holds the actual  
 ; starting address of message1.

PUTS ; print the message starting at address x3A00

; some more code

HALT

; shortened data block *(all the long strings have been moved out)*

...

message1\_ptr .FILL x3A00

...

.ORIG x3A00

.STRINGZ "this is a message" ; no longer needs a label

.END