; read two numbers using a subroutine and store them to memory
.ORIG x3000
; starting address is x3000
JSR READNUM
; read two numbers and store them
ST R0,NUM1
JSR READNUM
ST R0,NUM2
HALT

; subroutine developed as an extension of the
; earlier binary code

; read a decimal number from the keyboard,
; convert it from ASCII to 2's complement, and
; return it in R0. If any non-numeric character
; is pressed, or the number overflows, print an
; error message and start over.

; R0 holds the value of the last key pressed
; R1 holds the current value of the number being input
; R2 holds the additive inverse of ASCII '0' (0xFFD0)
; R3 is used as a temporary register

READNUM
; the subroutine to read a number
ST R7,SAVE_R7
; TRAP overwrites R7, so must save
ST R3,SAVE_R3
ST R2,SAVE_R2
ST R1,SAVE_R1
LD R2,NEG_0
; put the value -x30 in R2
AND R1,R1,#0
; clear the current value

READ_LOOP
GETC
; read a character
OUT
; echo it back to monitor
ADD R3,R0,#-10
; compare with ENTER
BRz DONE
; if ENTER pressed, done

ADD R0,R0,R2
; subtract x30 from R0
BRnz BAD_KEY
; smaller than '0' means error
ADD R3,R0,#-10
; check if > '9'
BRzp BAD_KEY
; greater than '9' means error
ADD R3,R1,R1
; sequence of adds multiplies R1 by 10
BRnz OVERFLOW
; overflow, but not really necessary here
ADD R3,R3,R3
BRnz OVERFLOW
; overflow, but not really necessary here
ADD R1,R1,R3
BRnz OVERFLOW
; overflow
ADD R1,R1,R1
BRnz OVERFLOW
; overflow
ADD R1,R1,RO
; finally, add in new digit
BRnz OVERFLOW
; overflow
BRnzp READ_LOOP
; get another digit

DONE
ADD R0,R1,#0
; move R1 into R0
LD R1,SAVE_R1
; restore register values for caller
LD R2,SAVE_R2
LD R3,SAVE_R3
LD R7,SAVE_R7
RET

; print error message: "non-digit pressed"
BAD_KEY
LEA R0,BK_MSG
; point R0 to the start of the string
PRINT_ERR
LEA R0,PRINT_ERR
; the trap that you're not allowed to use in MP2
Puts
AND R1,R1,#0
; reset current value
BRnzp READ_LOOP
; try reading again

; print error message: "overflow"
OVERFLOW
LEA R0,OF_MSG
; point R0 to the start of the string
BRnzp PRINT_ERR

SAVE_R1 .BLKW 1
; storage for saved register values
SAVE_R2 .BLKW 1
SAVE_R3 .BLKW 1
NEG_0 .FILL xFFD0
; the additive inverse of ASCII '0'
NUM1 .BLKW 1
; storage for the results
NUM2 .BLKW 1

; error messages. The sequence \n means newline and is replaced
; with a single ASCII linefeed character (\10). Similar sequences
; include \r for \13 (carriage return), \t for \9 (TAB), \ for
; backslash, etc.
BK_MSG .STRINGZ "\non-digit pressed\n"
OF_MSG .STRINGZ "\noverflow\n"

.END