Write, assemble and run successfully on the simulator a Mac-1 subroutine \( \text{lgneg}(n,x) \) that returns in the AC the address of the integer possessing the algebraically largest negative value along the real line among the \( n \) integers in the array whose starting address is \( x \). The largest negative value on the real line is the farthest right value to the left of zero. If there are no negative values among the \( n \) elements to be processed, then return -1 which is equivalent to the (unsigned) address 65535, or if we strip off the high order 4 bits, it corresponds to address 4095, the output status register; clearly, neither is a valid memory address for an array element. If there are two or more array entries that equally satisfy the requirements, return the address of the one with the highest address. Your subroutine should be tested with the main program shown below, which defines how the parameters are passed.

Hand in a copy of the main program symbolic assembly listing, the subroutine symbolic assembly listing, the contents of (macro) memory after “load main sub” (i.e., of main.abs) before execution of the program, and the contents of memory after execution of the program. Highlight and comment upon the final answers. Specify what values are contained in the addresses specified by \( \text{ans1} \), \( \text{ans2} \), and \( \text{ans3} \).