Problem ST-7 (5 parts) **Reverse Engineering**

Consider the following MIPS code.

label	instruction	comment
	addi $1, $0, 16	# x
	addi $2, $0, 3	# y
	addi $3, $0, 4	# z
	addi $4, $0, 0	# sum
Loop:	addi $1, $1, -1	# x--
A:	sll $5, $3, 7	# z*128 = z*8*16
B:	sll $6, $2, 4	# y*16
C:	add $7, $5, $6	# $7: z*128 + y*16
	add $8, $7, $1	# $8: z*128 + y*16 + x
	sll $8, $8, 2	# scale by 4
	lw $7, Array($8)	# use as offset into Array
	add $4, $4, $7	# $4: running sum
	bne $1, $0, Loop	# continue loop if x > 0
	jr $31	# return to caller

Part A (4 points) What type of loop is this (e.g., for, while, do while)?

do while

Part B (4 points) How many iterations does the loop perform?

16

Part C (9 points) The loop is accessing a multi-dimensional array. How many columns (Lx) and rows (Ly) does it have? At has at least how many planes (Lz)?

Lx = 16 **Ly = 8** **Lz ≥ 5**

Part D (4 points) In terms of the multi-dimensional array, what is being summed up and placed in register $4 by the program?

running sum of all elements of row 3 (4th row), on plane 4 (5th plane).

Part E (4 points) How can this program be optimized to reduce the total number of instructions executed by 45 instructions (from 149 to 104 instructions)?

Move the three instructions labeled A, B, and C above the loop, since the expression they compute does not change on each loop iteration.