Problem PA-3 (3 parts) Function and Stack

Part A The program `Sort` calls a subroutine `Mystery` (defined on next page). Complete the program `Sort` by adding MIPS code to correctly preserve appropriate registers before the `jal` by pushing them on the stack and to restore them after the subroutine call. Assume `Mystery` can modify any registers, not just the ones modified in the code on the next page.

Sort:	addi $1, $0, 4	# init Array index to 1
OuterLoop:	lw $3, Array($1)	# load in current element X
	add $2, $1, $0	# reinitialize Array index
	addi $29, $29, -12 **sw $1, 0($29)** **sw $3, 4($29)** **sw $31, 8($29)**	**# make room on stack for 3 words** **# push $1** **# push $3** **# push $31**
	jal Mystery	# in: $2, $3; out: $2
	lw $1, 0($29) **lw $3, 4($29)** **lw $31, 8($29)** **addi $29, $29, 12**	**# pop $1** **# pop $3** **# pop $31** **# readjust stack pointer**
	sw $3, Array($2)	# store current value here
	addi $1, $1, 4	# move current to next element
	slti $5, $1, 400	# all elements inserted?
	bne $5, $0, OuterLoop	# if not, then continue
	addi $1, $0, 196	# point to 49th element
	lw $3, Array($1)	# load 49th element
	addi $1, $1, 4	# point to 50th element
	lw $4, Array($1)	# load 50th element
	add $3, $3, $4	# sum 49th and 50th
	sra $2, $3, 1	# compute median (average)
	jr $31	# return to caller

Now consider the following MIPS subroutine that takes two inputs: an array index **I** in $2 and an integer array element Value in $3. Its result is placed in $2. The label Array is the base address of array A.

Label	Instruction	Comment
Mystery:	**...**	**# L0**
InnerLoop:	**addi $4, $2, -4**	**# L1**
	lw $4, Array($4)	**# L2**
	slt $5, $3, $4	**# L3**
	beq $5, $0, Exit	**# L4**
	sw $4, Array($2)	**# L5**
	addi $2, $2, -4	**# L6**
	bne $2, $0, InnerLoop	**# L7**
Exit:	**jr $31**	**# L8**

Part B If $2 holds index I into array A, what does $4 hold at line **L3**?

$4 = A[I-1]

Part C Write C code that is equivalent to the InnerLoop body of the Mystery subroutine. Assume A is a globally defined array. *For maximum credit, choose the most appropriate loop construct and declare and initialize variables as needed.*

```
int A[] = {9, 33, -15,...};
int Mystery(int I, int Value){
  do {
    if (Value < A[I-1]){
      A[I] = A[I-1];
      I--;
    }
    else
      break;
  } while(I);
  return(I);
}
```