**Lab 6. Array and Pointer – Assigments**

**Họ và tên: Phạm Long Khánh**

**MSSV: 20225640**

**Assignment 1:**

.data

message1: .asciiz "The maximum sum: "

message2: .asciiz "The lenght of prefix sub\_array: "

A: .word -2, 6, -1, 3, -2

.text

# t1 = maximum-sum, t0 = the lenght of prefix sub-array of A

main: la $a0, A # a0 -> address of A[0]

addi $a1, $0, 5 # The lenght of this array = a1 = 5 (n = 5)

slt $t0, $0, $a1 # Check if the lenght of this array greater than zero

bne $t0, $0, function # if true then branch to function

j exit # else exit program

end\_main: addi $v0, $0, 56 # print maximum sum

la $a0, message1

add $a1, $0, $t1

syscall

addi $v0, $0, 56 # print the lenght of max\_sum prefix sub\_array

la $a0, message2

add $a1, $0, $t0

syscall

exit: addi $v0, $0, 10 # exit

syscall

#-------------------------------------------------------------------------

function: addi $t0, $0, 1 # init the lenght of sub\_array of A in which max sum reachs = 1

lw $t1, 0($a0) # set max\_sum of a certain sub\_array = A[0]

add $t2, $t1, $0 # set current\_sum = A[0]

xor $s0, $0, $0 # i = 0

LOOP: beq $s0, $a1, DONE # if i = n then DONE

addi $s0, $s0, 1 # i = i + 1

sll $t3, $s0, 2 # t3 = 4 \* i

add $a2, $a0, $t3 # a2 -> address of A[i]

lw $t4, 0($a2) # t4 = A[i]

add $t2, $t2, $t4 # current\_sum = current\_sum + A[i]

slt $t5, $t1, $t2 # check if max\_sum < current\_sum

beq $t5, $0, continue # if not true then continue

add $t1, $0, $t2 # else max\_sum = current\_sum

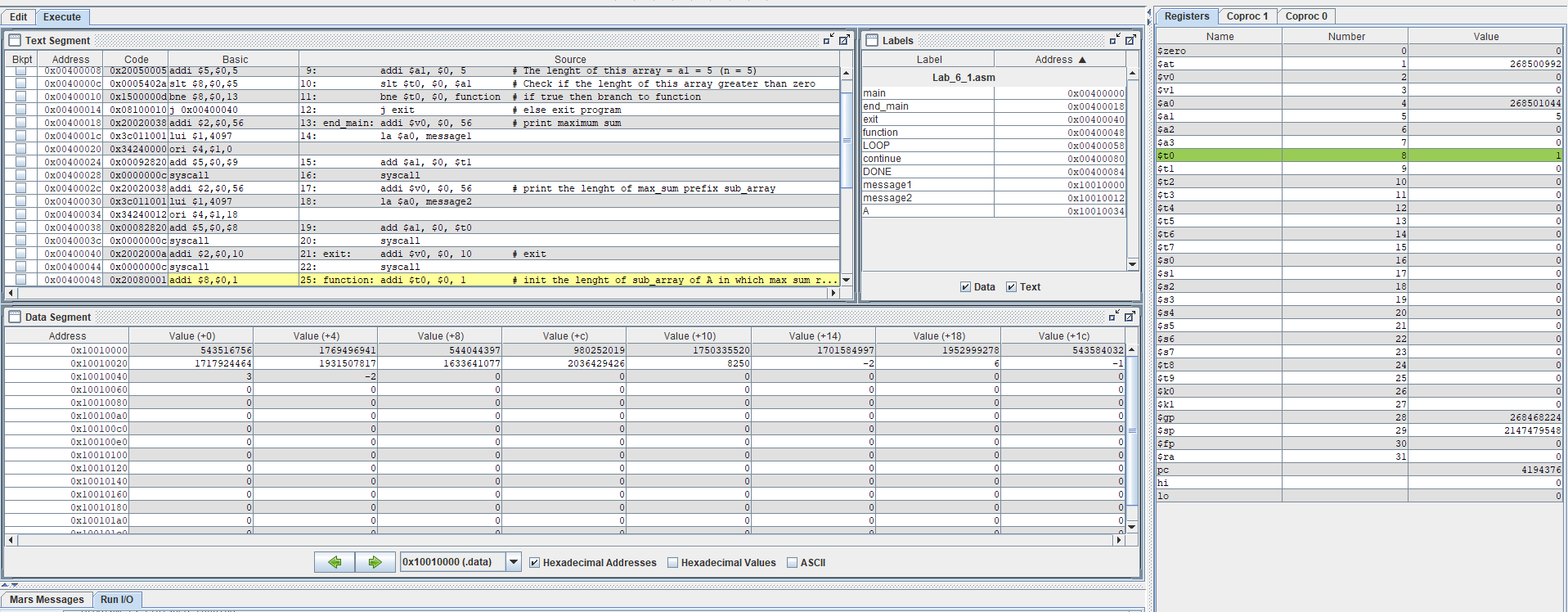
addi $t0, $s0, 1 # lenght of sub\_array = i + 1

continue:

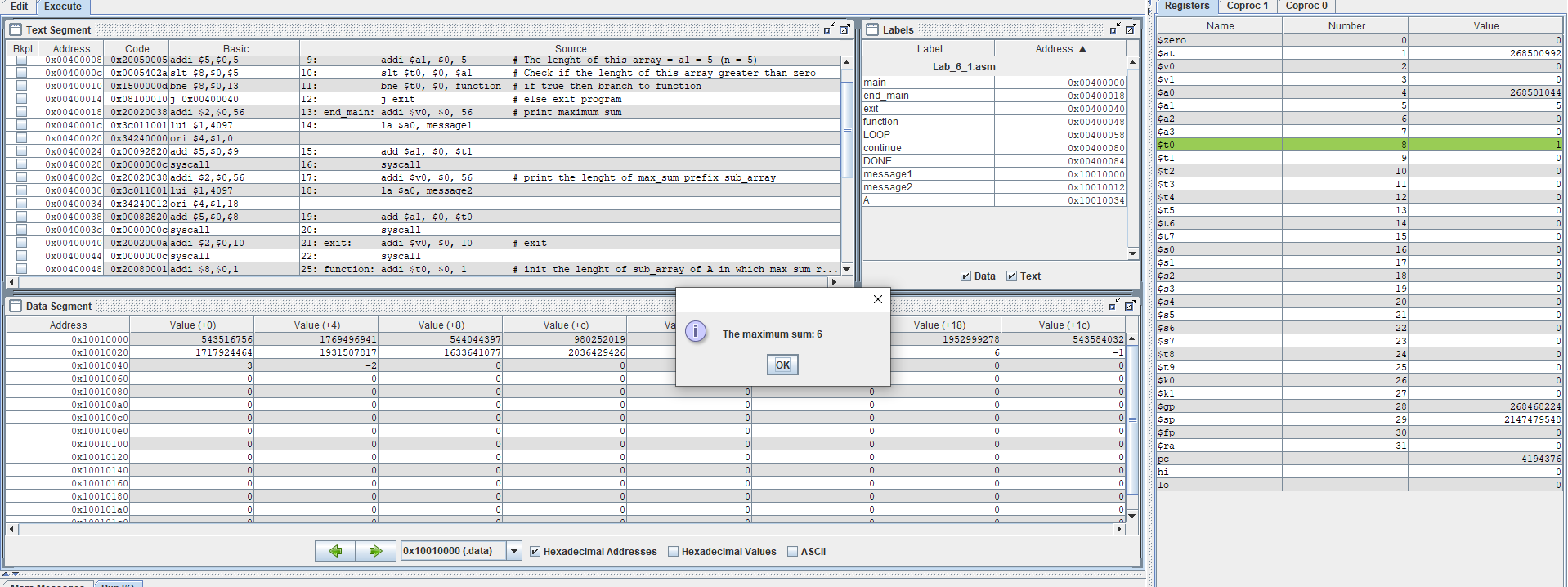
j LOOP

DONE: j end\_main

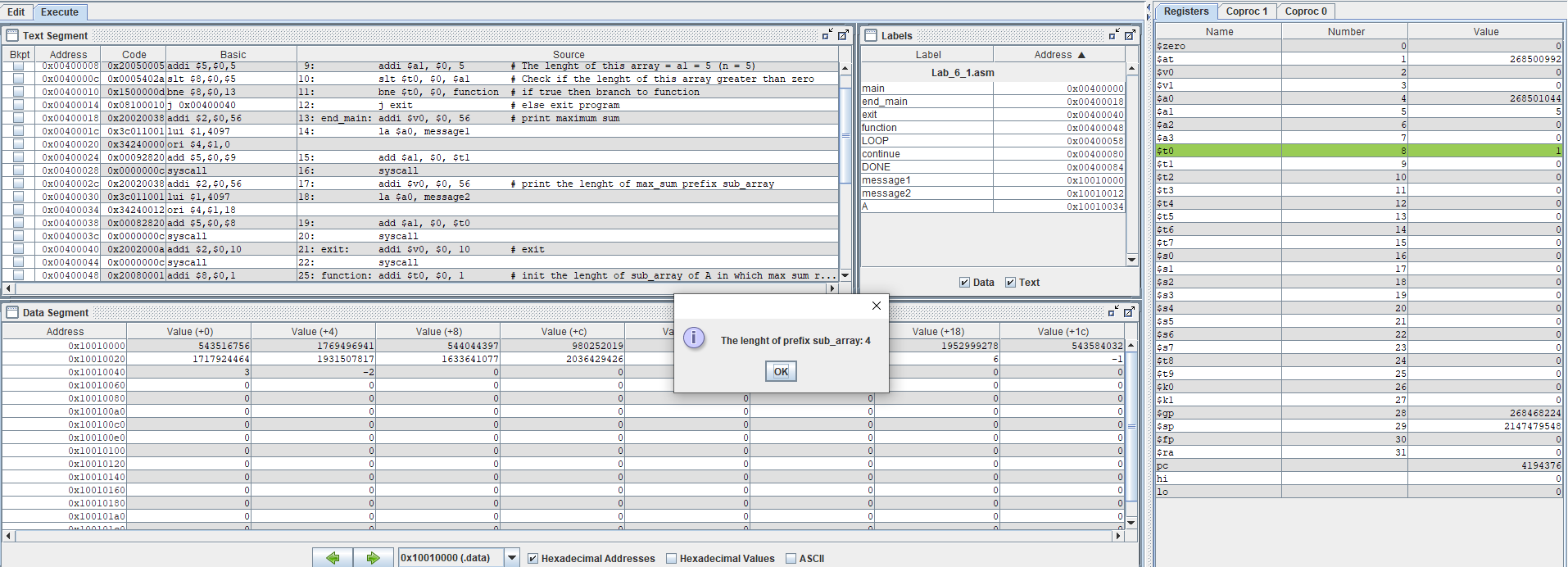
* Quá trình thực hiện chương trình:



* Kết quả của chương trình:
* Max prefix sum = 6



* The length of prefix sub array of A in which max sum reachs = 4
* Chương trình chạy đúng



**Assignment 2:**

.data

A: .word 7, -2, 5, 1, 5, 6, 7, 3, 6, 8, 8, 59, 5 # Array A has n elements (in this ex n = 13)

A\_end: .word # A\_end -> address of A[n]

.text

# Selection Sort (ascending sort using pointer)

main: la $a0, A # a0 -> address of A[0]

la $a1, A\_end

addi $a1, $a1, -4 # a1 -> address of A[n - 1]

j start\_sort

end\_sort: addi $v0, $0, 10 # exit

syscall

end\_main:

#----------------------------------------------------------------------------------------------

start\_sort: beq $a0, $a1, end\_sort # if list has 1 element then end\_sort

j find\_max # call the max procedure

after\_max: lw $s0, 0($a1) # load last element to s0

sw $s0, 0($t2) # copy last element to A[idx\_max] (idx\_max is a position that A[position] has max value in this list)

sw $t1, 0($a1) # copy max value in this list to A[n - 1]

addi $a1, $a1, -4 # decrement pointer to new last element

j start\_sort # repeat sort for smaller list with n - 1 elements

#----------------------------------------------------------------------------------------------

find\_max: addi $t0, $a0, 0 # set next pointer to first

lw $t1, 0($a0) # set max value to first value

addi $t2, $a0, 0 # set max pointer to first

LOOP: beq $t0, $a1, DONE # if next = last , done

addi $t0, $t0, 4 # point to next element

lw $s2, 0($t0) # get value of next element

sub $t3, $s2, $t1 # get subtraction of next element value and max value

blez $t3, LOOP # if next element value <= max, repeat

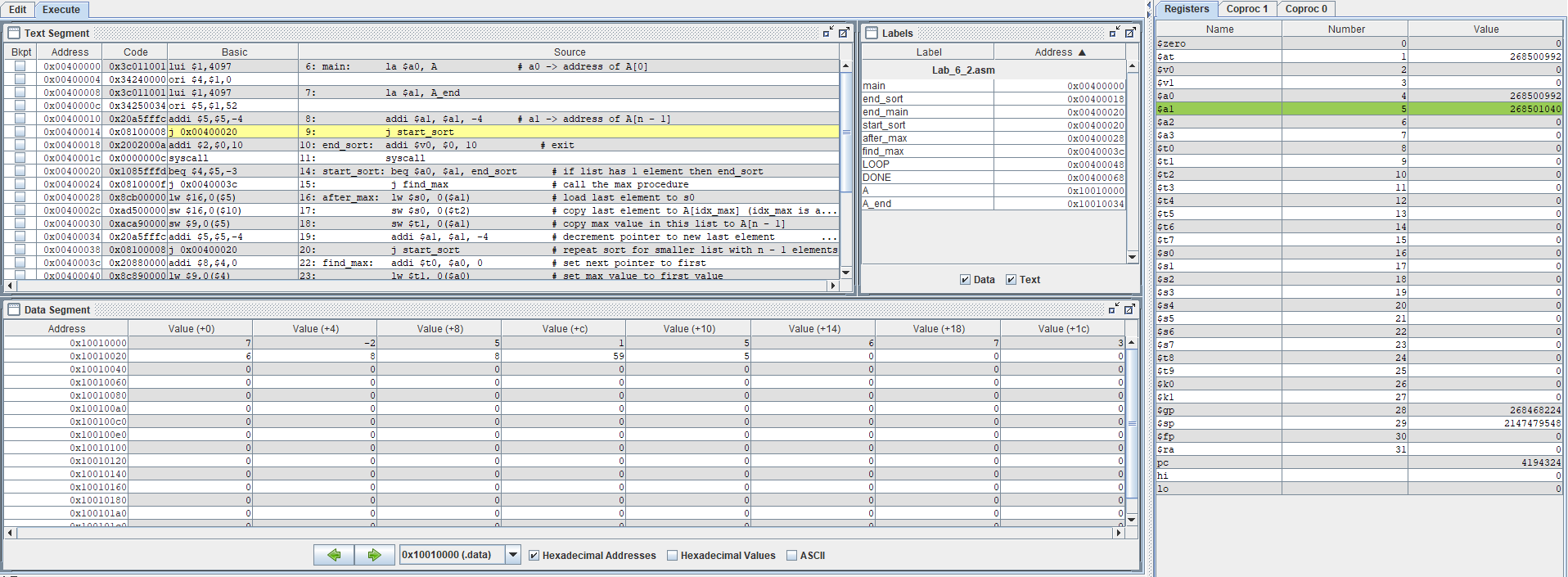
add $t1, $s2, $0 # else max value = next element value

add $t2, $t0, $0 # set max pointer to next element

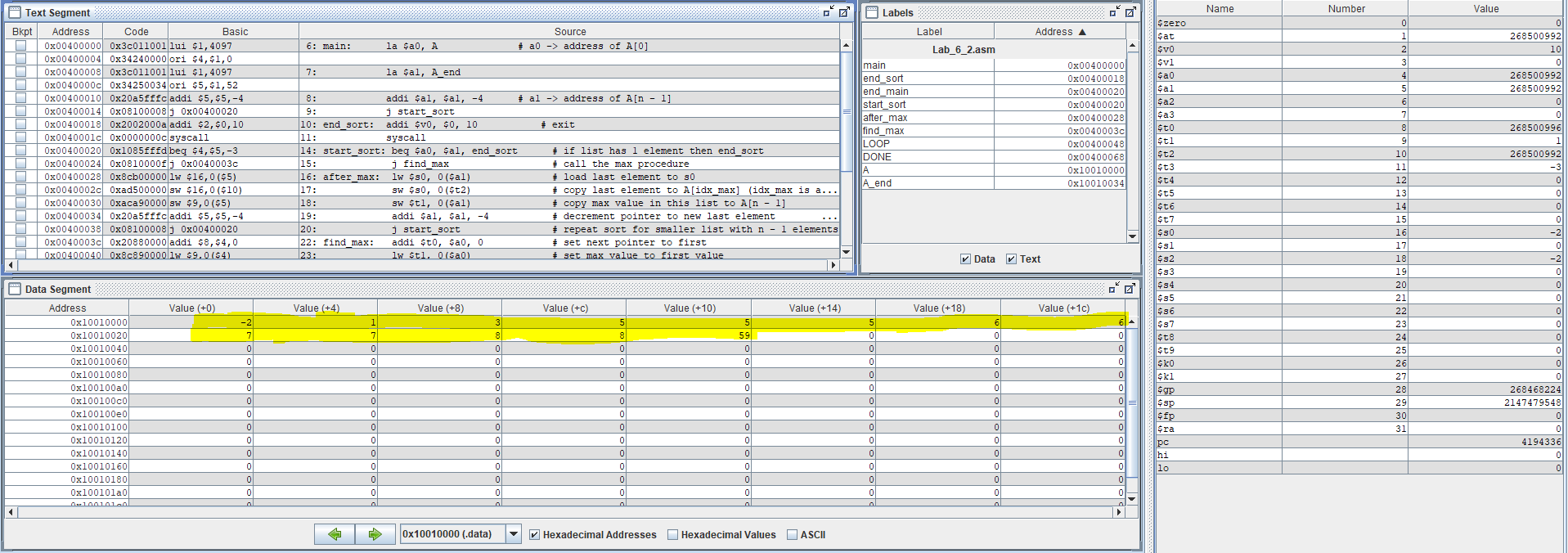
j LOOP # change completed; now repeat

DONE: j after\_max # Go to after\_max after finding the max value

* Quá trình thực hiện chương trình:



* Kết quả chương trình:
* Dãy sau khi sắp xếp:



* Chương trình chạy đúng

**Assignment 3:**

.data

A: .word 7, -2, 5, 1, 5, 6, 7, 3, 6, 8, 8, 59, 5 # Array A has n elements (in this ex n = 13)

A\_end: .word # A\_end -> address of A[n]

.text

# Bubble Sort (ascending sort using pointer)

main: la $a0, A # a0 -> address of A[0]

la $a1, A\_end

addi $a1, $a1, -4 # a1 -> address of A[n - 1]

j start\_sort # perform sort

end\_sort: addi $v0, $0, 10 # exit

syscall

end\_main:

#---------------------------------------------------------------------------------------------

start\_sort: beq $a0, $a1, end\_sort # if list has 1 element then end\_sort

j swap

after\_swap: addi $a1, $a1, -4 # decrement pointer to new last element

j start\_sort # repeat sort for samller list with n - 1 elements

#---------------------------------------------------------------------------------------------

swap: addi $t0, $a0, 0 # set t0 point to first

addi $v1, $0, 1 # set swapped = 0 (To check if this list has change after swap)

LOOP: beq $t0, $a1, DONE # if first = last, done

addi $t1, $t0, 4 # set t1 point to next element

add $t2, $t0, $0 # set t2 point to current element

addi $t0, $t1, 0 # set t0 point to next element

lw $s0, 0($t2) # s0 = A[i]

lw $s1, 0($t1) # s1 = A[i + 1]

sub $s2, $s0, $s1 # s2 = A[i] - A[i+1]

blez $s2, LOOP # if A[i] <= A[i + 1], repeat

sw $s0, 0($t1) # swap A[i] with A[i + 1]

sw $s1, 0($t2)

addi $v1, $0, 1 # This list has some changes

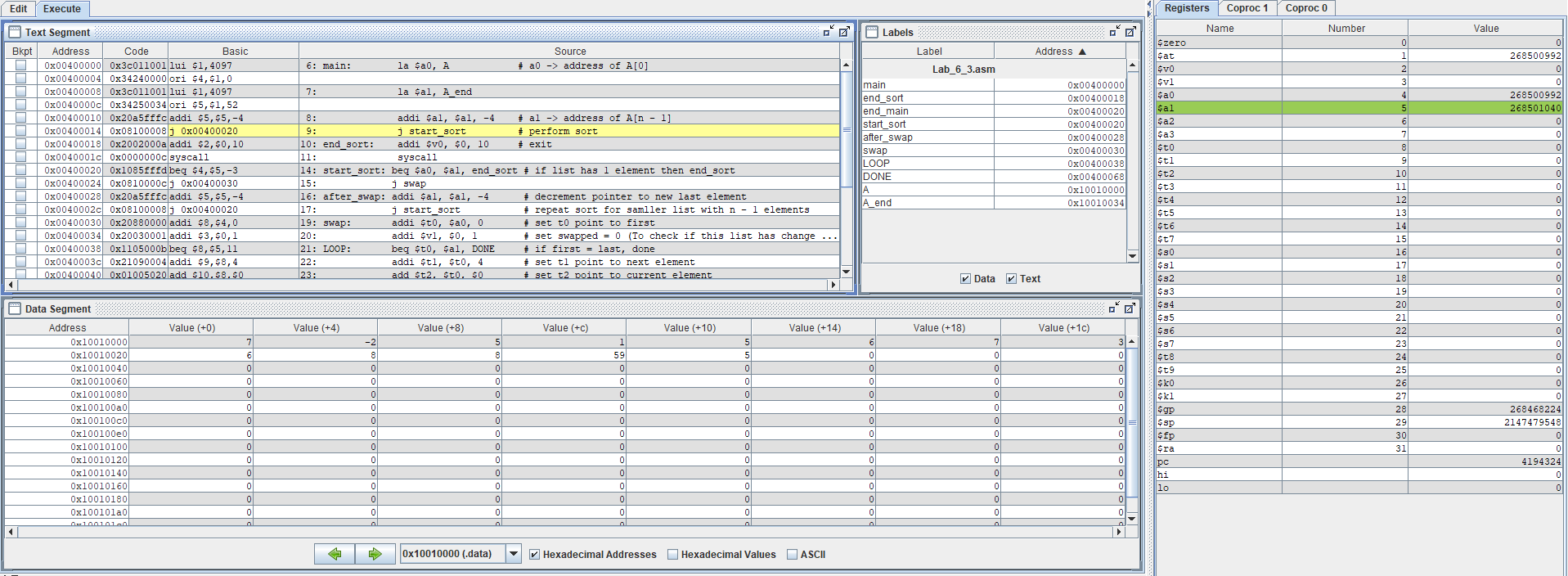
j LOOP

DONE: bne $v1, $0, after\_swap # if this list has some changes,

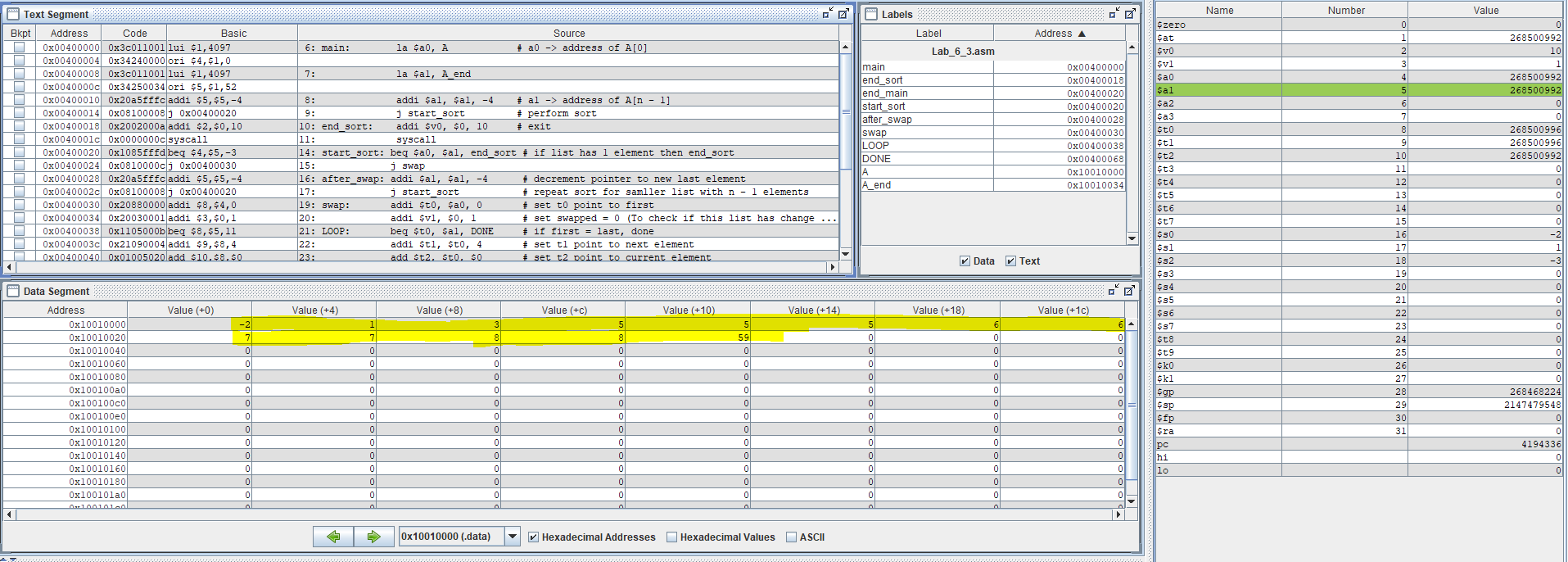
# continue sorting

j end\_sort # else finish sorting

* Quá trình chạy chương trình:



* Kết quả chương trình:
* Dãy sau khi sắp xếp:



* Chương trình chạy đúng

**Assignment 4:**

.data

A: .word 7, -2, 5, 1, 5, 6, 7, 3, 6, 8, 8, 59, 5 # Array A has n elements (in this ex n = 13)

A\_end: .word # A\_end -> address of A[n]

.text

# Insertion Sort (ascending sort using pointer)

main: la $a0, A # a0 -> address of A[0]

addi $a2, $a0, 0 # a2 -> address of A[0] (make a temporary pointer to keep address of A[0] in execution process)

la $a1, A\_end

addi $a1, $a1, -4 # a1 -> address of A[n - 1]

j start\_sort

end\_sort: addi $v0, $0, 10 # exit

syscall

#------------------------------------------------------------------------------

start\_sort: sub $t4, $a0, $a1 # t4 = i - end

bgtz $t4 end\_sort # if (i > end) then finish sorting

j insert

after\_insert: addi $a0, $a0, 4 # set a0 point to next element (i = i + 1)

j start\_sort # repeat sort for the element with index i

#------------------------------------------------------------------------------

insert: # a0 -> address of A[i]

add $t0, $a0, 0 # t0 -> address of A[j] (j = i)

lw $s0, 0($a0) # s0 = A[i]

LOOP: # while loop

beq $t0, $a2, end\_loop # if (j = 0) then end\_loop. Note: set this condition here to prevent t1 point to A[-1] (at this time j = 0)

addi $t1, $t0, -4 # set t1 point to A[j - 1]

lw $t2, 0($t1) # t2 = A[j - 1]

sub $t3, $t2, $s0 # t3 = A[j - 1] - A[i]

blez $t3, end\_loop # if (A[j - 1] > A[i]) then end\_loop

sw $t2, 0($t0) # A[j] = A[j - 1]

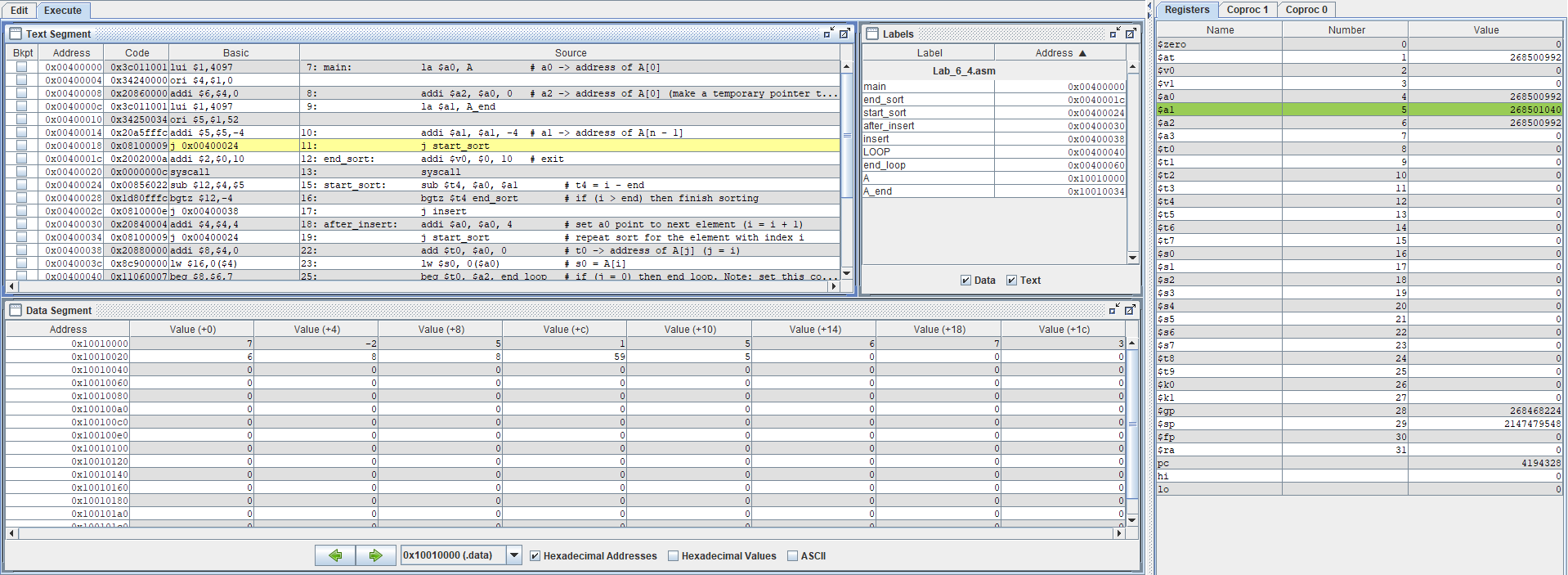
addi $t0, $t0, -4 # set t0 point to A[j - 1] (j = j - 1)

j LOOP # repeat until a suitable position is found

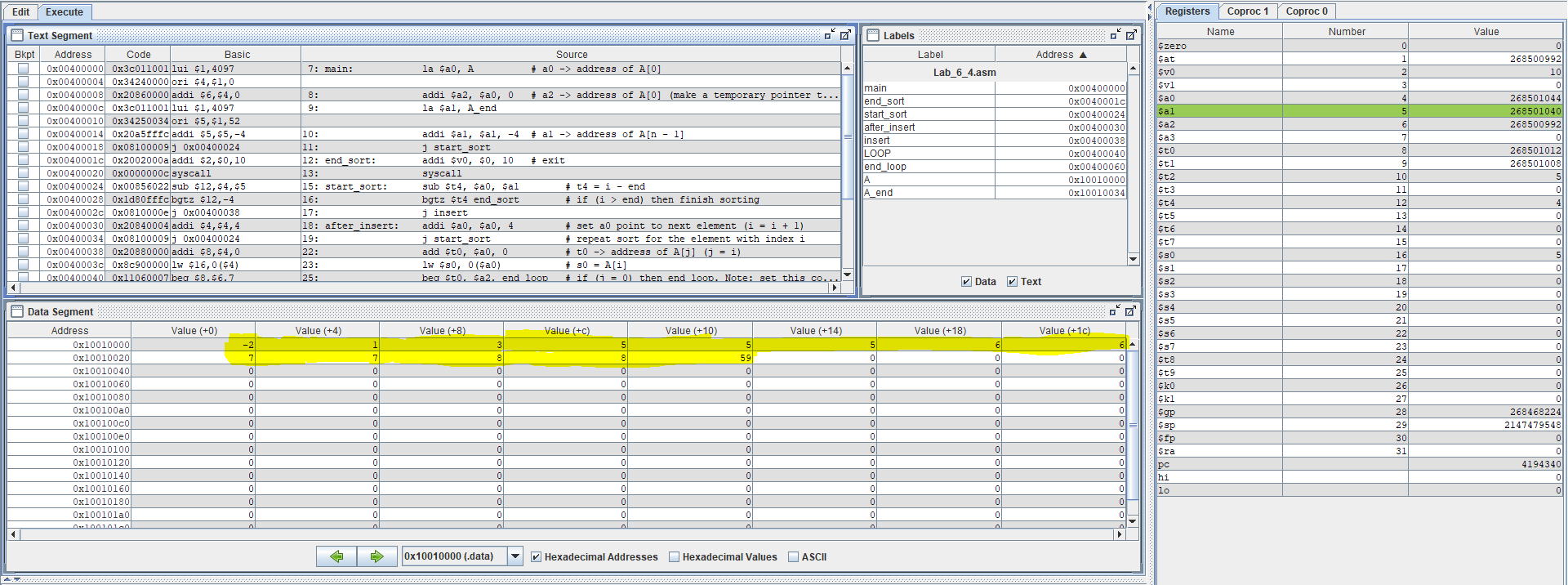
end\_loop: sw $s0, 0($t0) # A[j] = s0 = A[i] (insert successful)

j after\_insert

* Quá trình chạy chương trình:



* Kết quả chương trình:
* Dãy sau khi sắp xếp:



* Chương trình chạy đúng