**Thực hành kiến trúc máy tính tuần 6**

Giảng viên : Lê Bá Vui

Sinh viên : Hoàng Đức Anh

MSSV : 20176688

Assignment 1

Code:

.data

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

.text

main:

la $a0,A

li $a1,5

j mspfx

nop

continue:

lock: j lock

nop

end\_of\_main:

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

#Procedure mspfx

# @brief find the maximum-sum prefix in a list of integers

# @param[in] a0 the base address of this list(A) need to be processed

# @param[in] a1 the number of elements in list(A)

# @param[out] v0 the length of sub-array of A in which max sum reachs.

# @param[out] v1 the max sum of a certain sub-array

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

#Procedure mspfx

#function: find the maximum-sum prefix in a list of integers

#the base address of this list(A) in $a0 and the number of

#elements is stored in a1

mspfx: addi $v0,$zero,0 #initialize length in $v0 to 0

addi $v1,$zero,0 #initialize max sum in $v1to 0

addi $t0,$zero,0 #initialize index i in $t0 to 0

addi $t1,$zero,0 #initialize running sum in $t1 to 0

loop: add $t2,$t0,$t0 #put 2i in $t2

add $t2,$t2,$t2 #put 4i in $t2

add $t3,$t2,$a0 #put 4i+A (address of A[i]) in $t3

lw $t4,0($t3) #load A[i] from mem(t3) into $t4

add $t1,$t1,$t4 #add A[i] to running sum in $t1

slt $t5,$v1,$t1 #set $t5 to 1 if max sum < new sum

bne $t5,$zero,mdfy #if max sum is less, modify results

j test #done?

mdfy:

addi $v0,$t0,1 #new max-sum prefix has length i+1

addi $v1,$t1,0 #new max sum is the running sum

test: addi $t0,$t0,1 #advance the index i

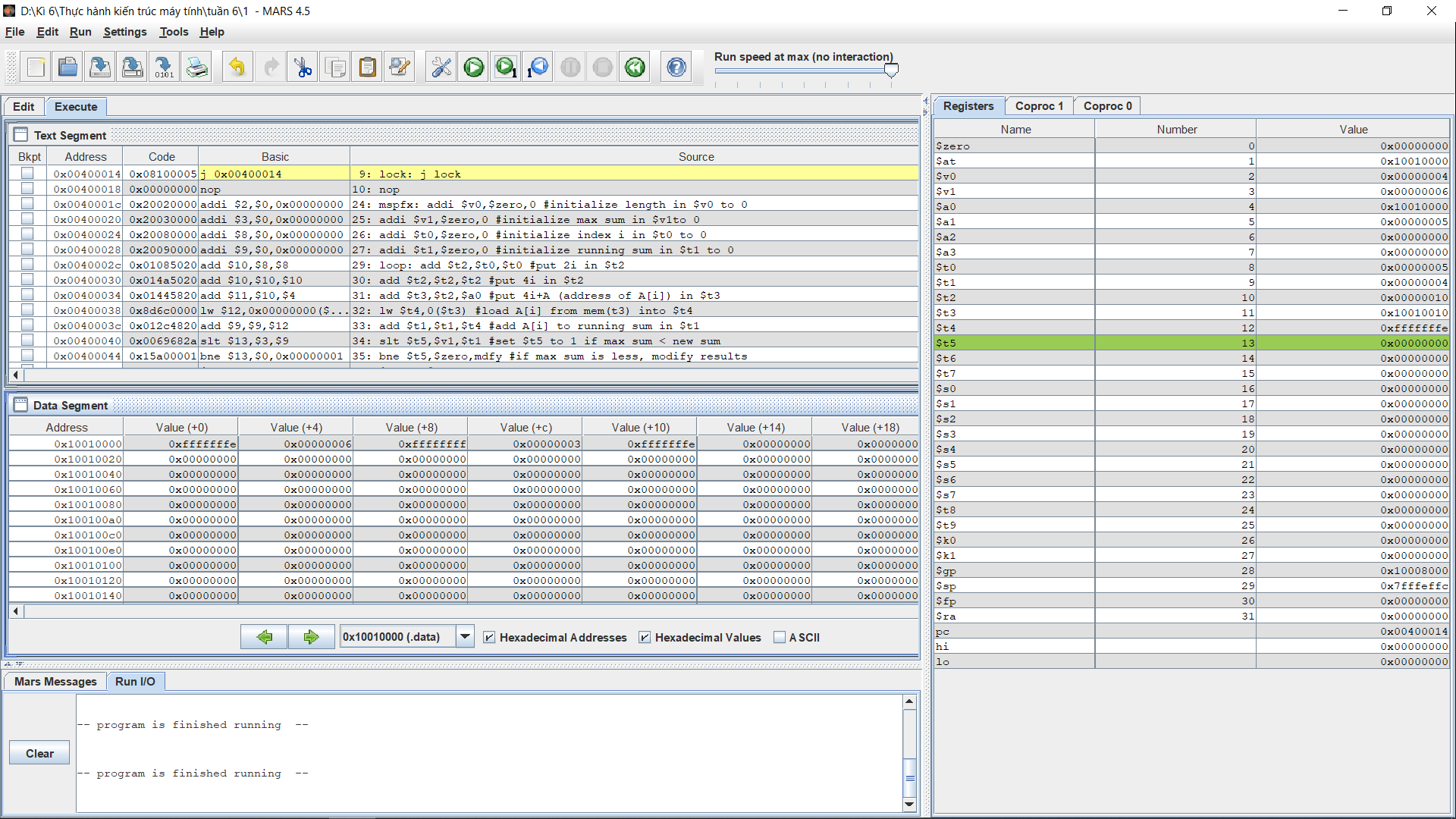
slt $t5,$t0,$a1 #set $t5 to 1 if i<n

bne $t5,$zero,loop #repeat if i<n

done: j continue

mspfx\_end:

Kết quả





Kết quả là chính xác do đầu vào là: { -2, 6, -1, 3, -2}

Sub lớn nhất là 6 ứng với $v1 và độ dài của mảng đến khi có tổng lớn nhất là 4 ứng với $v0

Assignment 2

Code:

.data

A: .word 7, -2, 5, 1, 5,6,7,3,6,8,8,59,5

Aend: .word

.text

main: la $a0,A #$a0 = Address(A[0])

la $a1,Aend

addi $a1,$a1,-4 #$a1 = Address(A[n-1])

j sort #sort

after\_sort: li $v0, 10 #exit

syscall

end\_main:

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

#procedure sort (ascending selection sort using pointer)

#register usage in sort program

#$a0 pointer to the first element in unsorted part

#$a1 pointer to the last element in unsorted part

#$t0 temporary place for value of last element

#$v0 pointer to max element in unsorted part

#$v1 value of max element in unsorted part

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

sort: beq $a0,$a1,done #single element list is sorted

j max #call the max procedure

after\_max: lw $t0,0($a1) #load last element into $t0

sw $t0,0($v0) #copy last element to max location

sw $v1,0($a1) #copy max value to last element

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

j sort #repeat sort for smaller list

done: j after\_sort

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

#Procedure max

#function: fax the value and address of max element in the list

#$a0 pointer to first element

#$a1 pointer to last element

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

max:

addi $v0,$a0,0 #init max pointer to first element

lw $v1,0($v0) #init max value to first value

addi $t0,$a0,0 #init next pointer to first

loop:

beq $t0,$a1,ret #if next=last, return

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

lw $t1,0($t0) #load next element into $t1

slt $t2,$t1,$v1 #(next)<(max) ?

bne $t2,$zero,loop #if (next)<(max), repeat

addi $v0,$t0,0 #next element is new max element

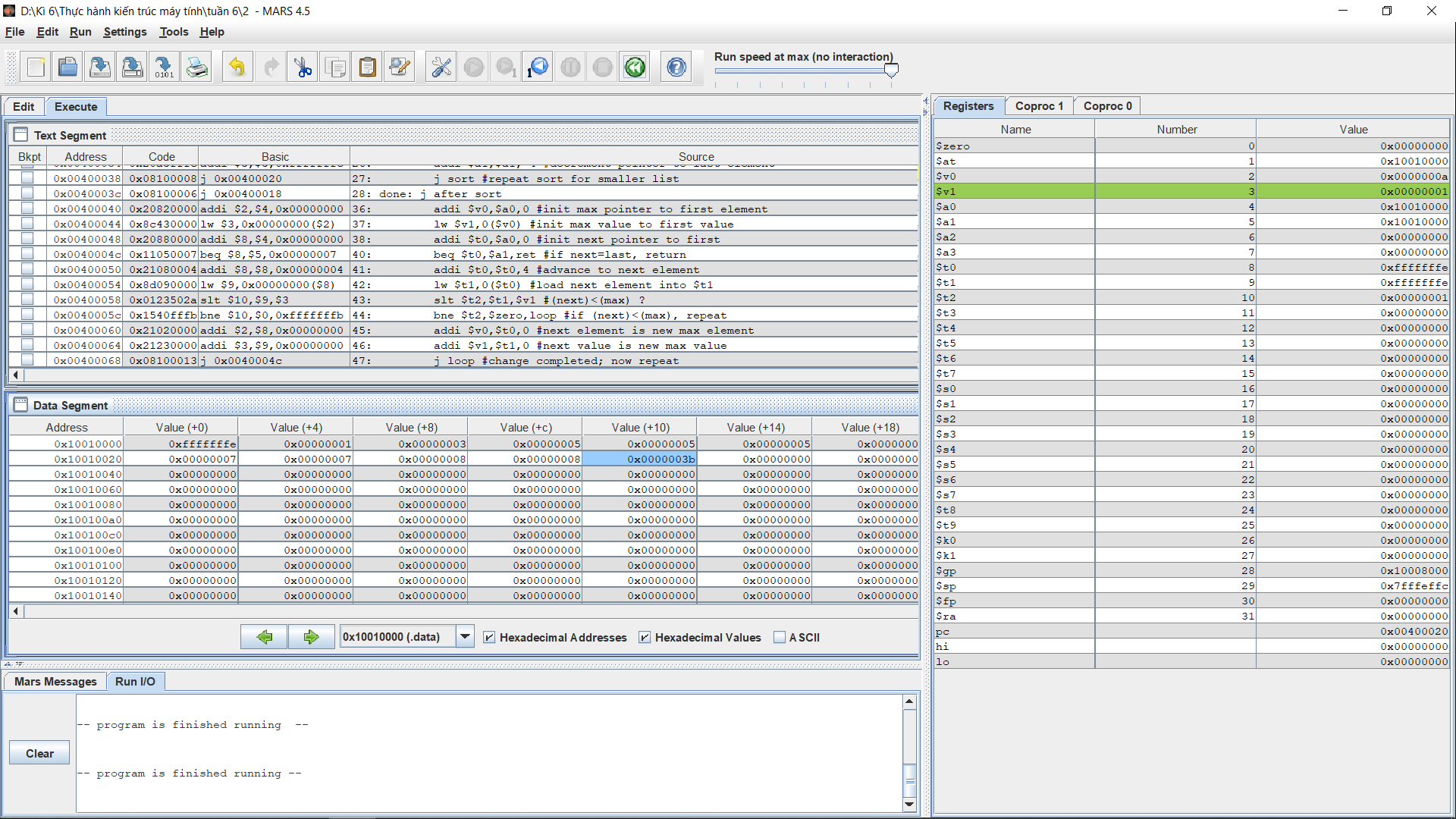
addi $v1,$t1,0 #next value is new max value

j loop #change completed; now repeat

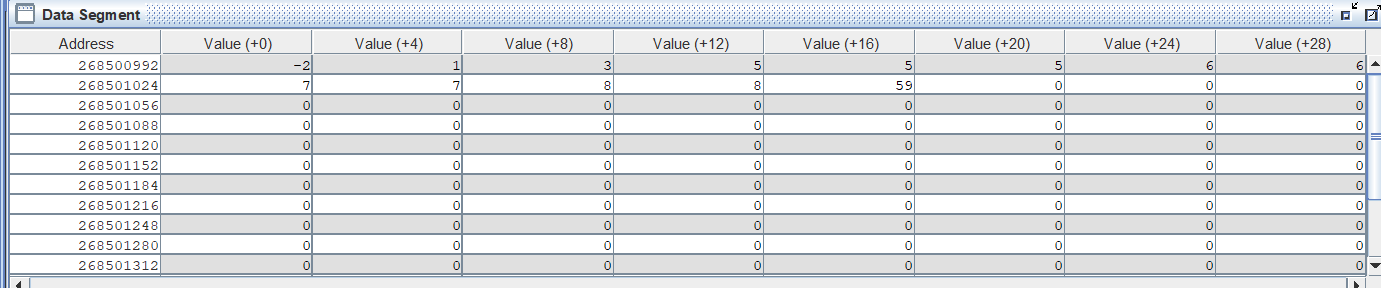
ret:

j after\_max

Kết quả:



Dầu vào là mảng { 7, -2, 5, 1, 5,6,7,3,6,8,8,59,5} kết quả in ra:



Kết quả chính xác

Assignment 3

Code:

.data

.align 4

Table: .space 24

msg1: .asciiz "Please insert an integer: "

msg2: .asciiz " "

msg3: .asciiz "\nOutput: "

.text

.globl main

main:

addi $s0,$zero,5

addi $t0,$zero,0

in: # input

li $v0,4

la $a0,msg1

syscall

li $v0,5

syscall

add $t1,$t0,$zero

sll $t1,$t0,2

add $t3,$v0,$zero

sw $t3,Table ( $t1 )

addi $t0,$t0,1

slt $t1,$s0,$t0

beq $t1,$zero,in

la $a0,Table

addi $a1,$s0,1 #a1=6 #call buble\_sort

jal buble\_sort #print table

li $v0,4

la $a0,msg3

syscall

la $t0,Table

#s0=5

add $t1,$zero,$zero

printtable: #print Input

lw $a0,0($t0)

li $v0,1

syscall

li $v0,4

la $a0,msg2

syscall

addi $t0,$t0,4

addi $t1,$t1,1

slt $t2,$s0,$t1

beq $t2,$zero,printtable

li $v0,10

syscall

buble\_sort:

#a0=address of table

#a1=sizeof table

add $t0,$zero,$zero #counter1( i )=0

loop1:

addi $t0,$t0,1 #i++

bgt $t0,$a1,endloop1 #if t0 < a1 break;

add $t1,$a1,$zero #counter2=size=6

loop2:

bge $t0,$t1,loop1 #j < = i

#slt $t3,$t1,$t0

#bne $t3,$zero,loop1

addi $t1,$t1,-1 #j--

mul $t4,$t1,4 #t4+a0=table[j]

addi $t3,$t4,-4 #t3+a0=table[j-1]

add $t7,$t4,$a0 #t7=table[j]

add $t8,$t3,$a0 #t8=table[j-1]

lw $t5,0($t7)

lw $t6,0($t8)

bgt $t5,$t6,loop2 #switch t5,t6

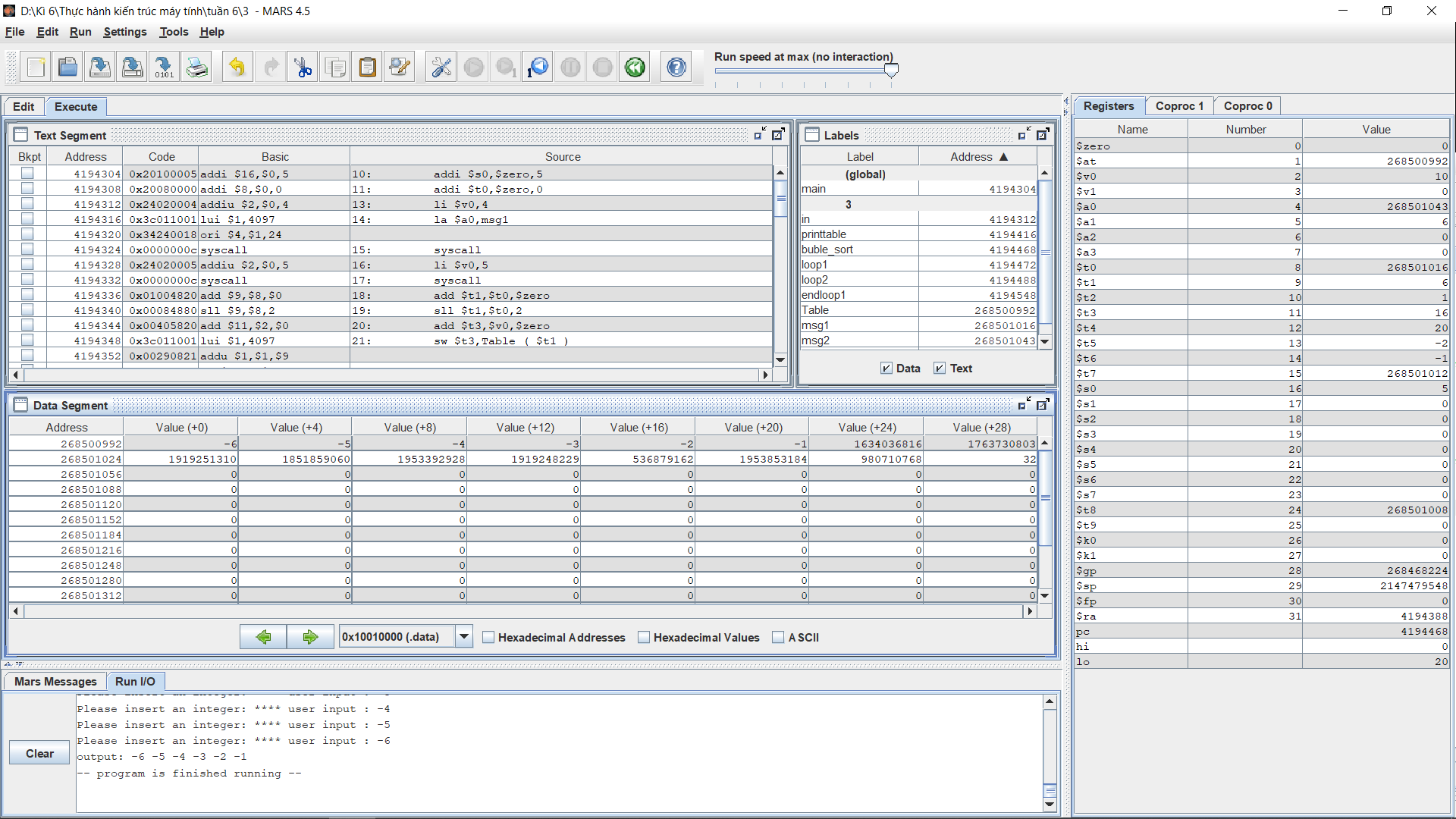
sw $t5,0($t8)

sw $t6,0($t7)

j loop2

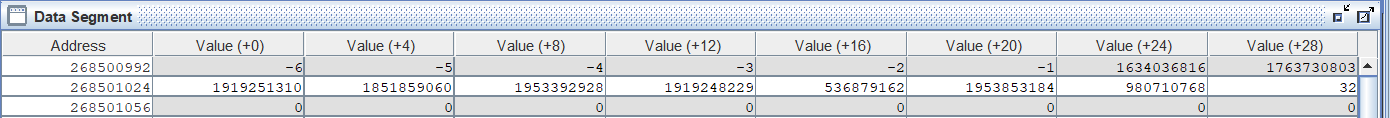
endloop1:

jr $ra



Đầu vào là mảng:{-1, -2, -3, -4, -5, -6}

Kết quả in ra:



Assignment 4

Code

.data

A: .word -1, -2, 4, 5, -6, -3, 10, 9, 8, 7, 100

length: .word 10

.text

main:

# Use $v0 to hold firstUnsortedIndex

# Use $v1 to hold testIndex

# Use $a0 to hold elementToInsert

# Use $a1 to hold value of A[ .. ]

# Use $a2 to calculate the address of A[ ... ] in

# Use $a3 to hold the value of (length-1)

# Use $t0 to hold the base/starting address of the A array

for\_init: li $v0, 1

lw $a3, length

sub $a3, $a3, 1

la $t0, A

for\_loop: bgt $v0, $a3, end\_for

sub $v1, $v0, 1

mul $a2, $v0, 4 # address of A[i]= base addr of A + i\*(element size)

add $a2, $t0, $a2

lw $a0, 0($a2)

while: blt $v1, 0, end\_while

mul $a2, $v1, 4 # address of A[i]= base addr of A + i\*(element size)

add $a2, $t0, $a2

lw $a1, 0($a2)

ble $a1, $a0, end\_while

sw $a1, 4($a2)

sub $v1, $v1, 1

j while

end\_while:

mul $a2, $v1, 4 # address of numbers[i]= base addr of numbers + i\*(element size)

add $a2, $t0, $a2

sw $a0, 4($a2)

addi $v0, $v0, 1

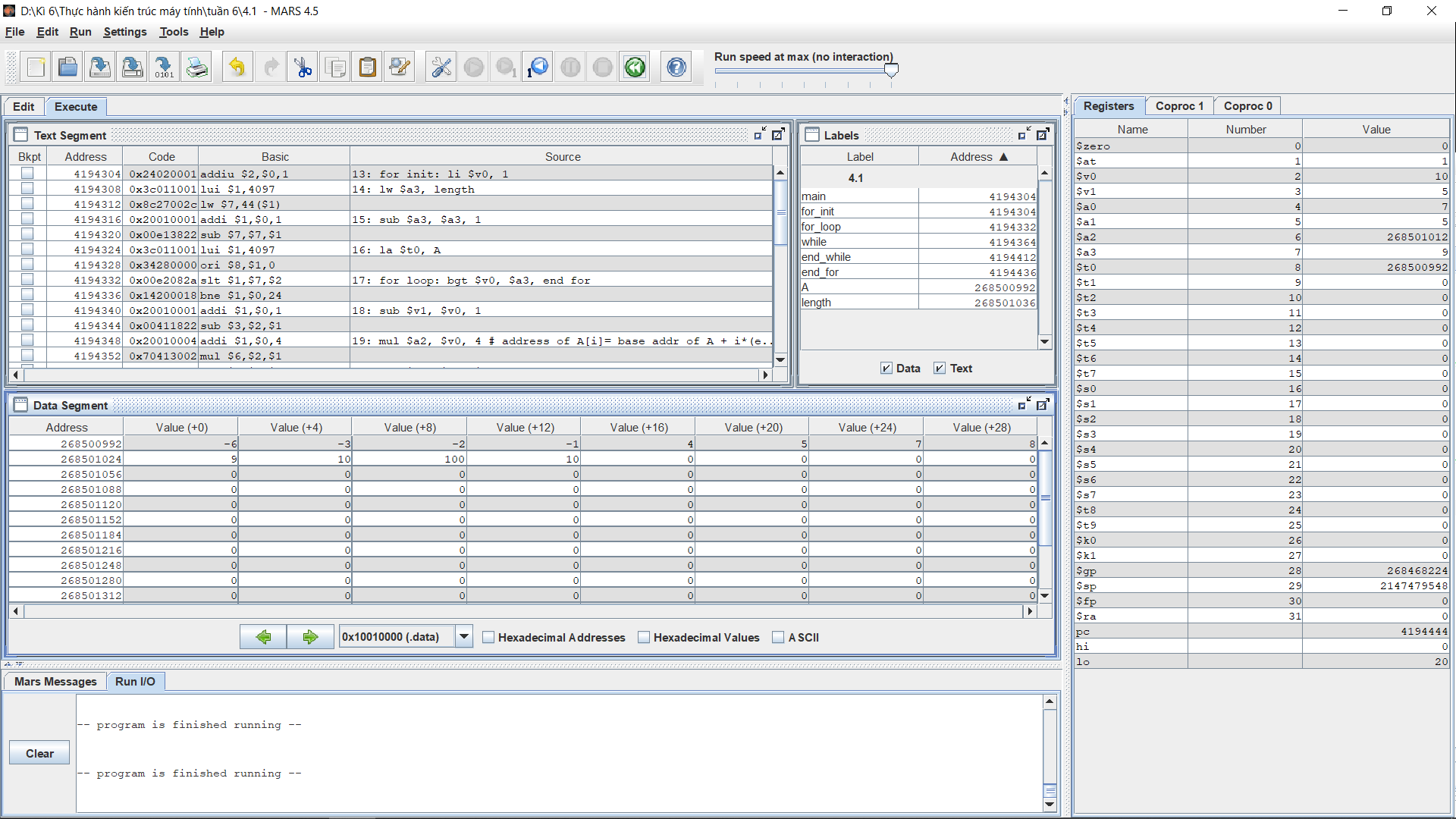
j for\_loop

end\_for:

li $v0, 10 # system call to exit

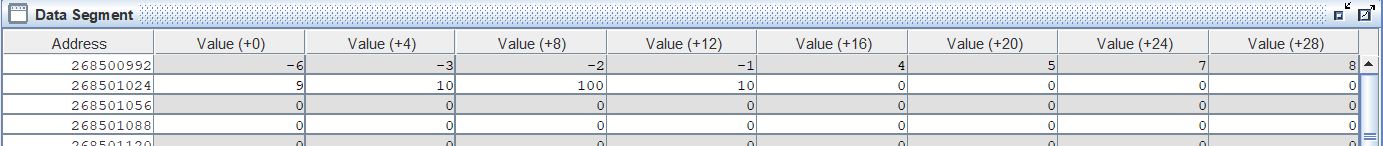
syscall

Kết quả



Đầu vào: {-1, -2, 4, 5, -6, -3, 10, 9, 8, 7, 100}

Kết quả in ra:



**Conclusion**

* + Lợi ích:
    - Dùng chỉ mục trực quan dễ hiểu.
    - Dùng con trỏ tiết kiệm tính toán.
  + Hạn chế:
    - Dùng chỉ mục cần tốn thời gian tính toán địa chỉ.
    - Dùng con trỏ khó khiểm soát và không trực quan dễ hiểu.