BÀI TẬP TUẦN 7

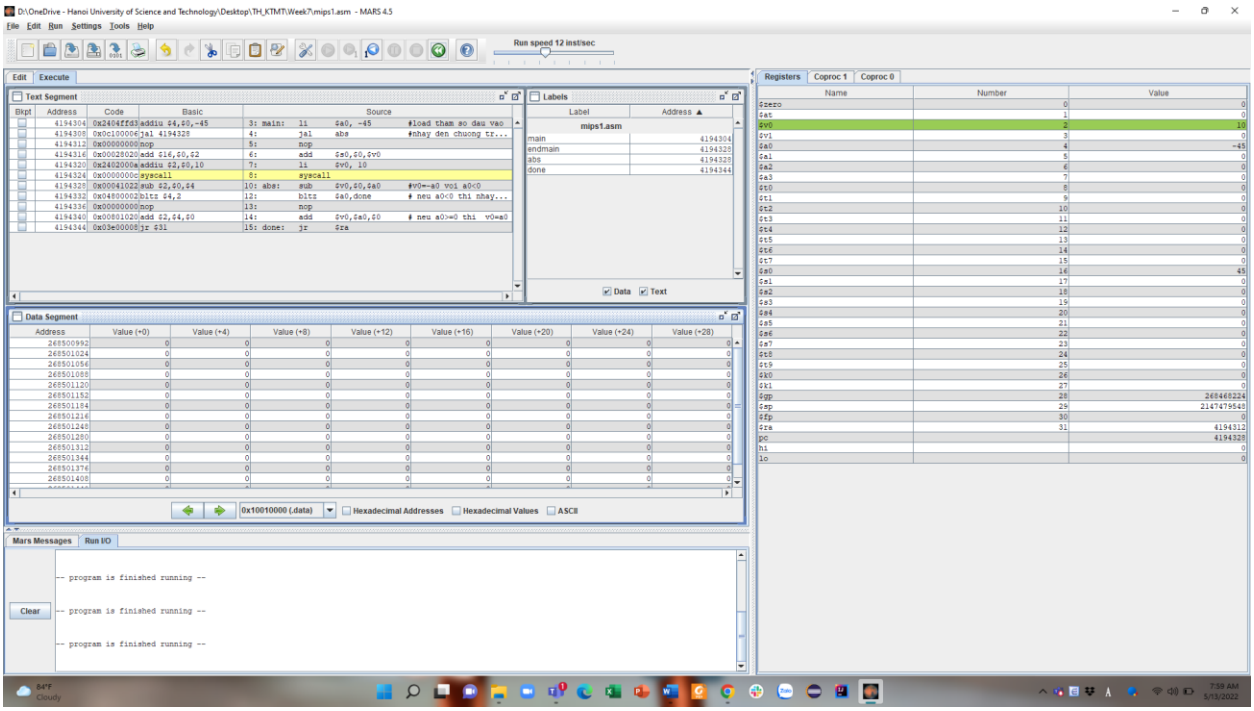
Exercise 1:

- *Chương trình:*

```
#Laboratory Exercise 7 Home Assignment 1
.text
main:   li      $a0, -45        #load tham so dau vao
        jal     abs             #nhay den chuong trinh con abs va luu dia chi tra ve trong thanh ghi $ra
        nop
        add     $s0,$0,$v0
        li      $v0, 10
        syscall
endmain:
abs:    sub     $v0,$0,$a0      #v0=-(a0) voi (a0)<0

        bltz    $a0,done        # neu (a0)<0 thi nhay den done
        nop
        add     $v0,$a0,$0      # neu (a0)>=0 thi  v0=(a0)
done:   jr      $ra
```

- *Kết quả:*

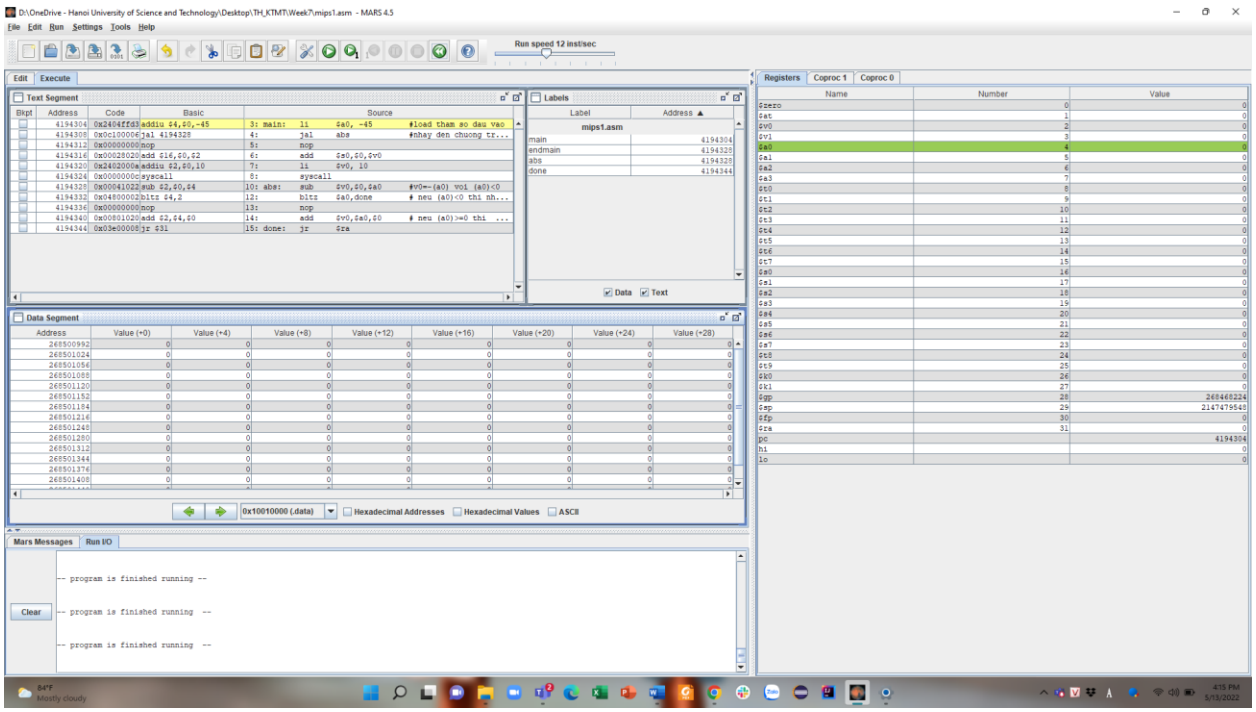


- Giải thích:

Trước khi chạy đến chương trình con abs:

+ $ra là 0. ($ra = 0)

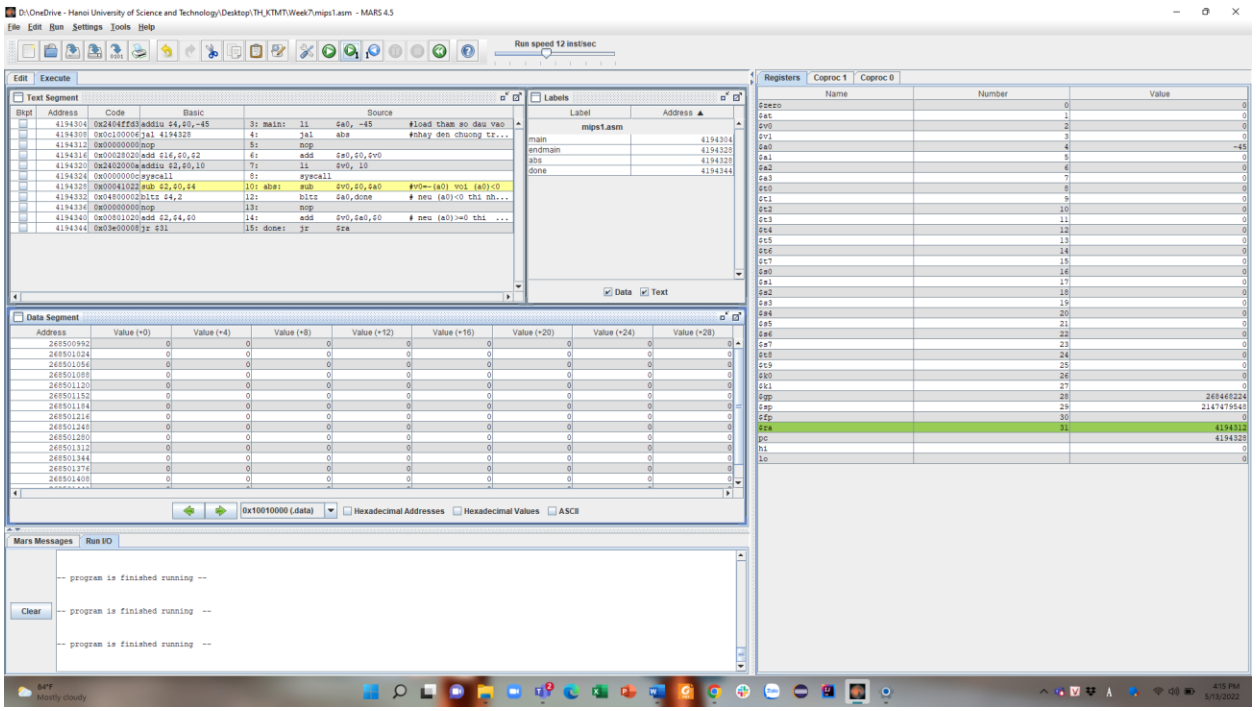
+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194304)



Sau khi chạy đến chương trình con abs:

+ $ra ghi địa chỉ của câu lệnh liền sau câu lệnh jal là nop.($ra = 4194312)

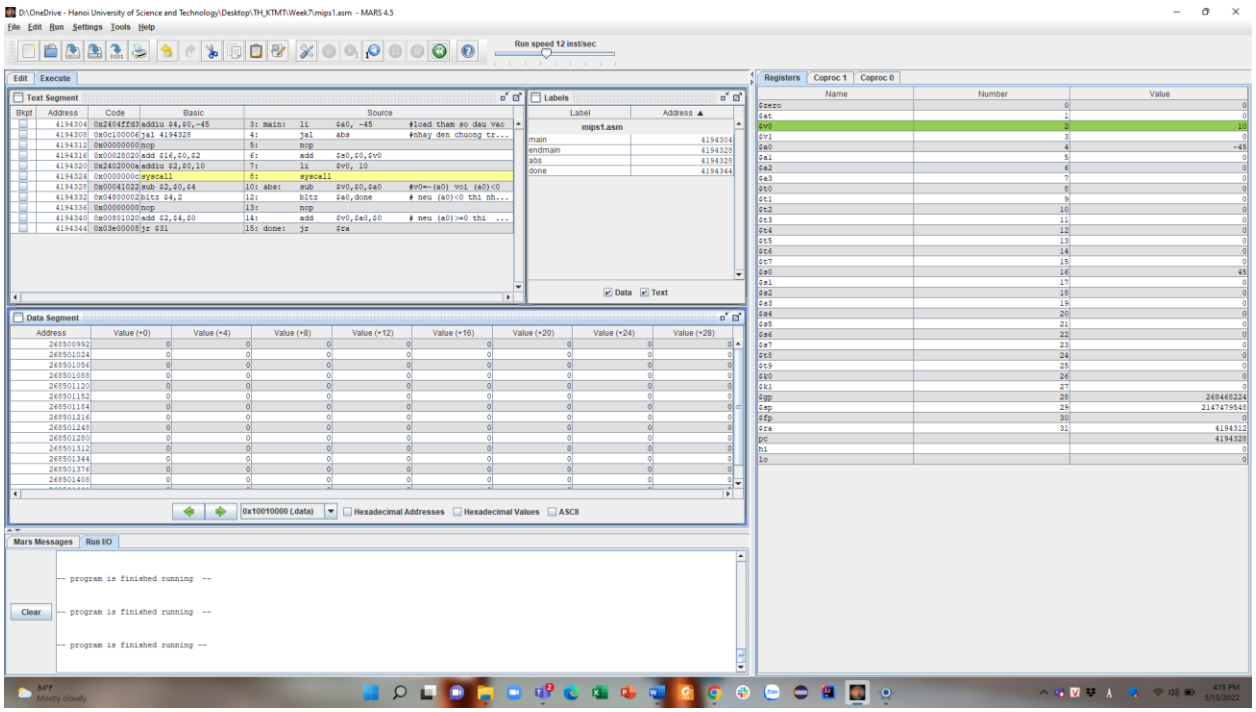
+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194328)



Sau khi kết thúc chương trình:

+ $ra ghi địa chỉ của câu lệnh liền sau câu lệnh jal. ($ra = 4194312)

+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194328)

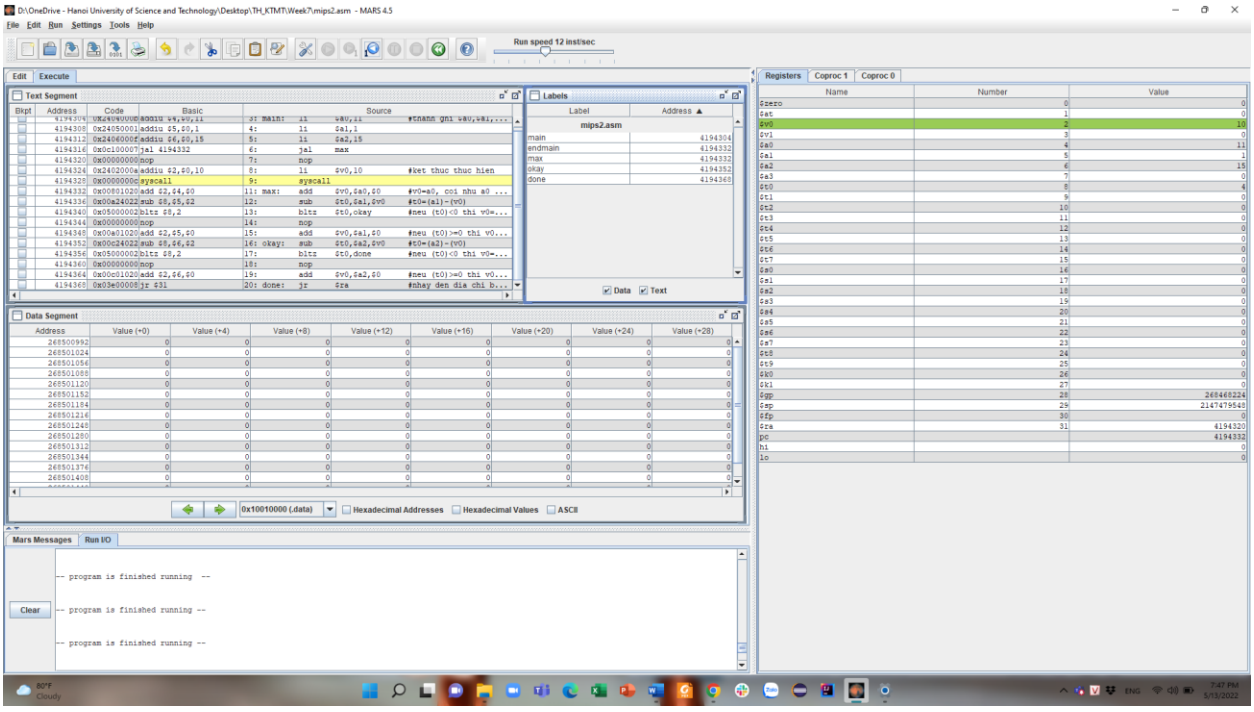


Exercise 2:

- *Chương trình:*

```
#Laboratory Exercise 7, Home Assignment 2
.text
main:   li      $a0,11          #thanh ghi $a0,$a1,$a2 luu cac so de test
        li      $a1,1
        li      $a2,15
        jal     max
        nop
        li      $v0,10          #ket thuc thuc hien
        syscall
endmain:
max:    add     $v0,$a0,$0      #v0=a0, coi nhu a0 la so lon nhat
        sub     $t0,$a1,$v0     #t0=(a1)-(v0)
        bltz    $t0,okay        #neu (t0)<0 thi v0=(a0)
        nop
        add     $v0,$a1,$0      #neu (t0)>=0 thi v0=(a1), coi nhu a1 la so lon nhat cho den hien tai
okay:   sub     $t0,$a2,$v0     #t0=(a2)-(v0)
        bltz    $t0,done        #neu (t0)<0 thi v0=(a0)
        nop
        add     $v0,$a2,$0      #neu (t0)>=0 thi v0=(a2)
done:   jr      $ra             #nhay den dia chi ben trong thanh ghi $ra
```

- *Kết quả:*

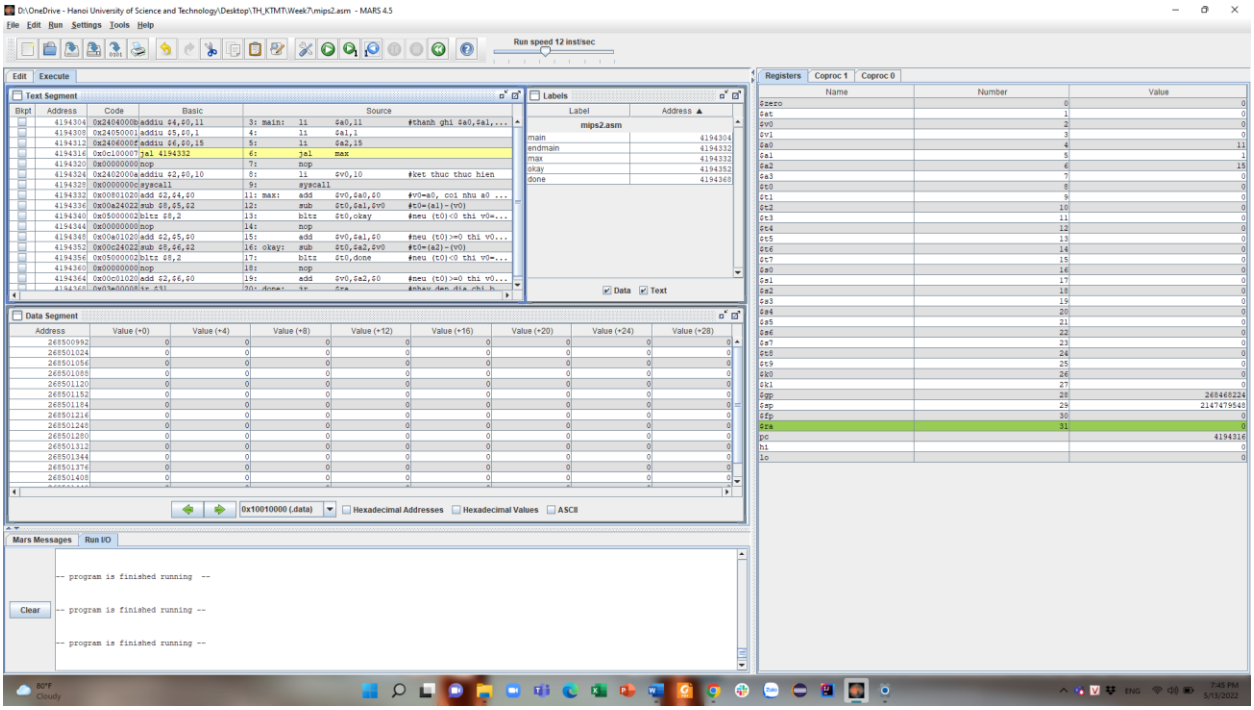


- Giải thích:

Trước khi chạy đến chương trình con max:

+ $ra là 0. ($ra = 0)

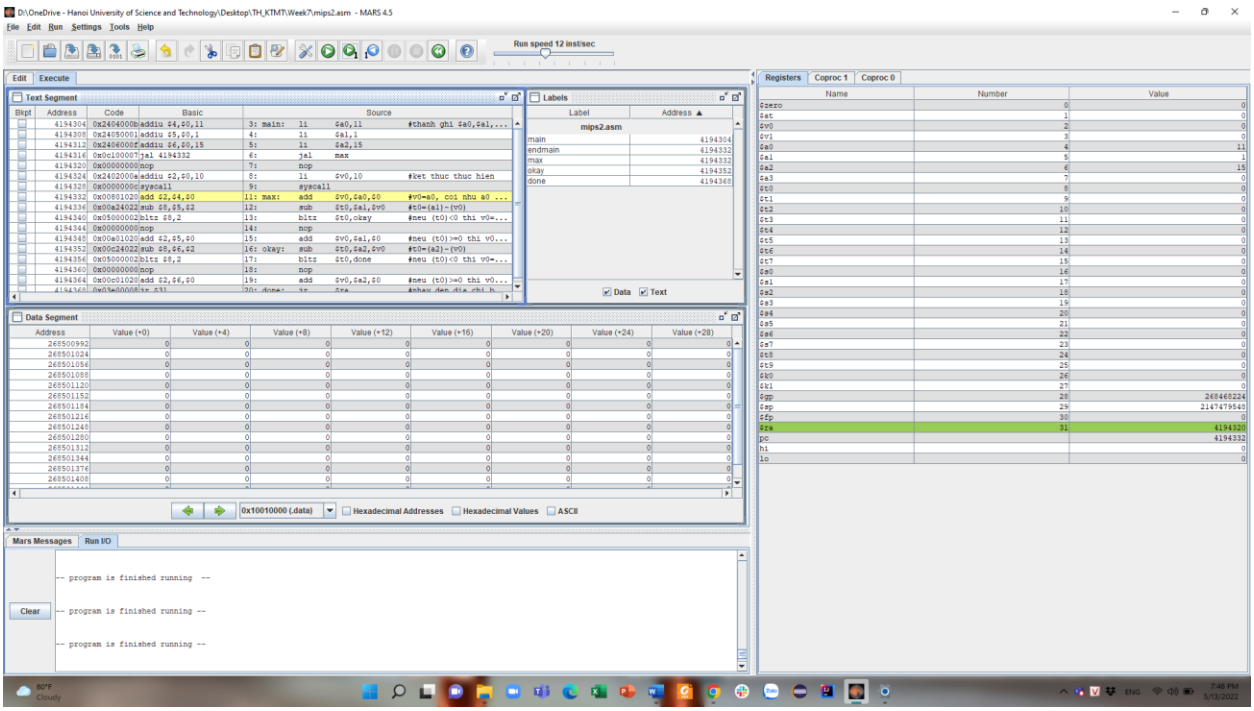
+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194316)



Sau khi chạy đến chương trình con max:

+ $ra ghi địa chỉ của câu lệnh liền sau câu lệnh jal là nop. ($ra = 4194320)

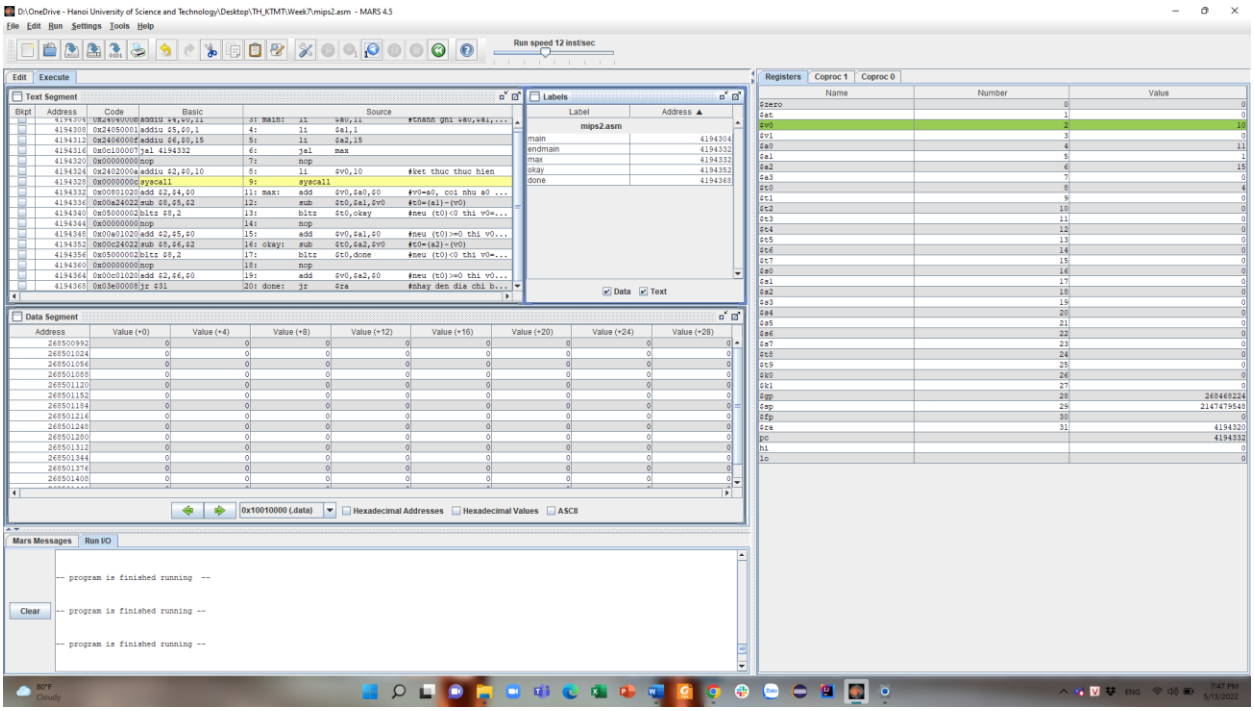
+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194332)



Sau khi kết thúc chương trình:

+ $ra ghi địa chỉ của câu lệnh liền sau câu lệnh jal là nop. ($ra = 4194320)

+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194332)

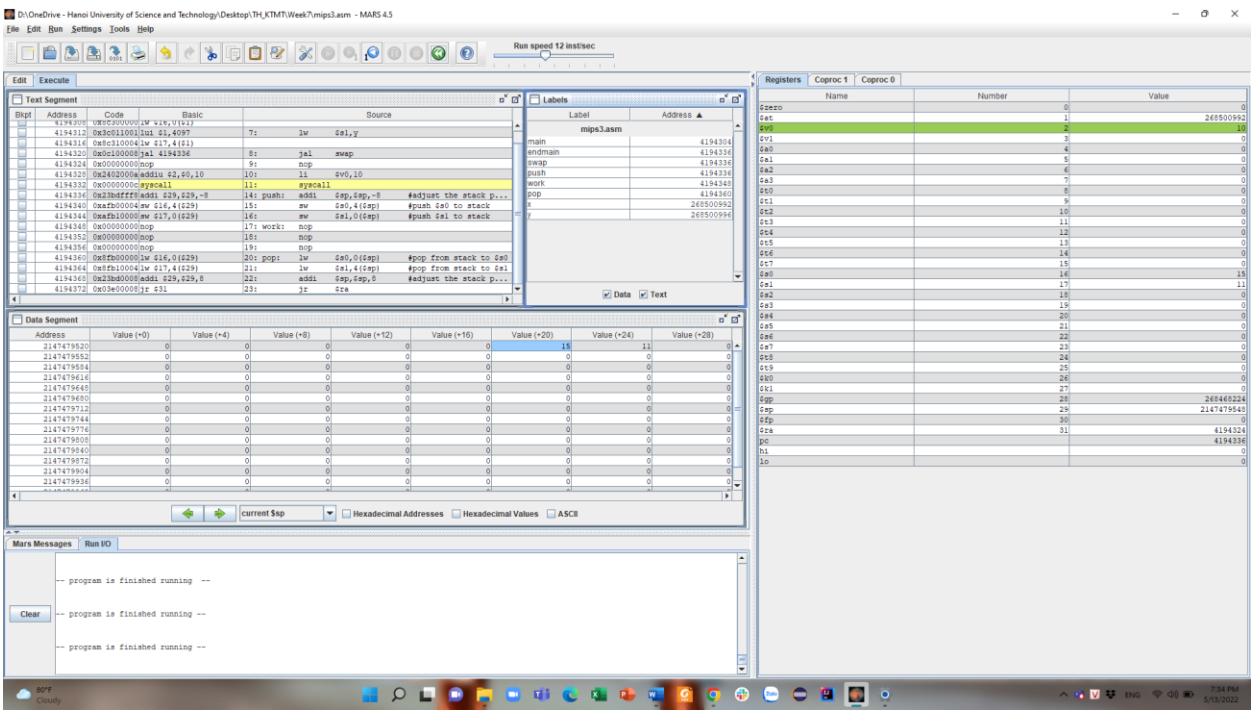


Exercise 3:

- *Chương trình:*

```
#Laboratory Exercise 7, Home Assignment 3
.data
x:      .word   11              #khai bao x,y
y:      .word   15
.text
main:   lw      $s0,x
        lw      $s1,y
        jal     swap
        nop
        li      $v0,10
        syscall
endmain:
swap:
push:   addi    $sp,$sp,-8      #adjust the stack pointer
        sw      $s0,4($sp)      #push $s0 to stack
        sw      $s1,0($sp)      #push $s1 to stack
work:   nop
        nop
        nop
pop:    lw      $s0,0($sp)      #pop from stack to $s0
        lw      $s1,4($sp)      #pop from stack to $s1
        addi    $sp,$sp,8       #adjust the stack pointer
        jr      $ra
```

- *Kết quả:*

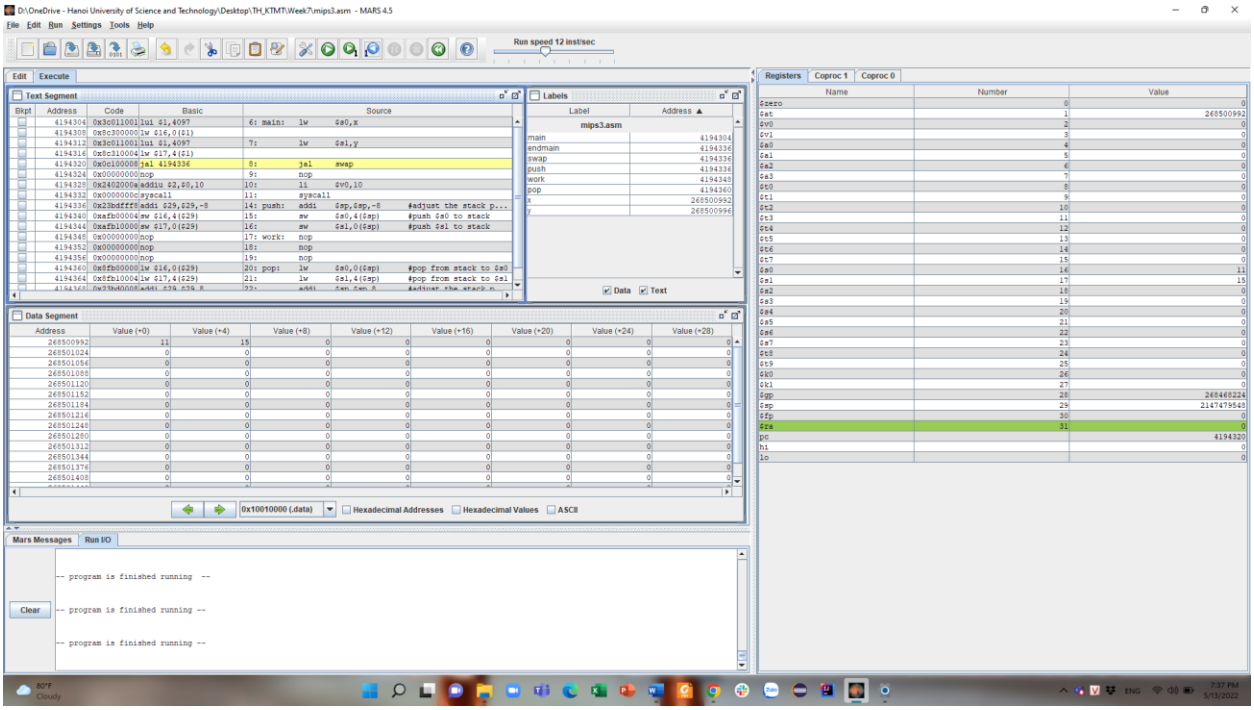


- Giải thích:

Trước khi chạy đến chương trình con swap:

+ $ra là 0. ($ra = 0)

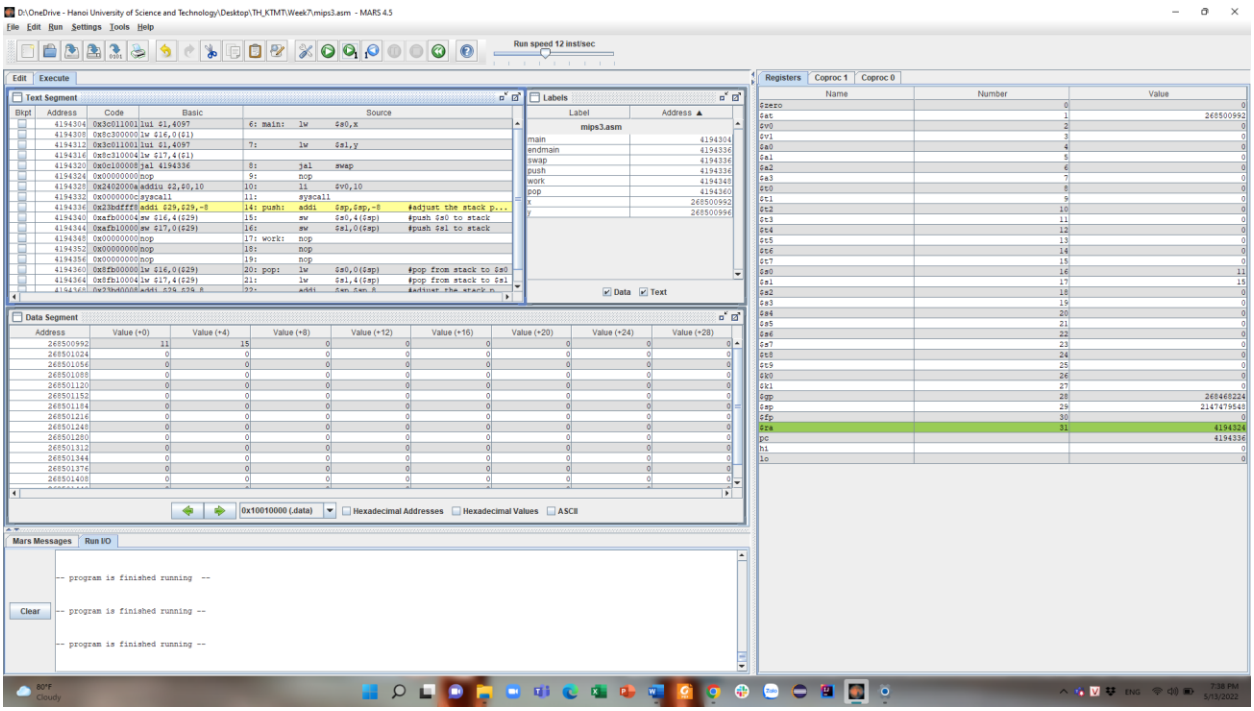
+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194320)



Sau khi chạy đến chương trình con swap:

+ $ra ghi địa chỉ của câu lệnh liền sau câu lệnh jal là nop. ($ra = 4194324)

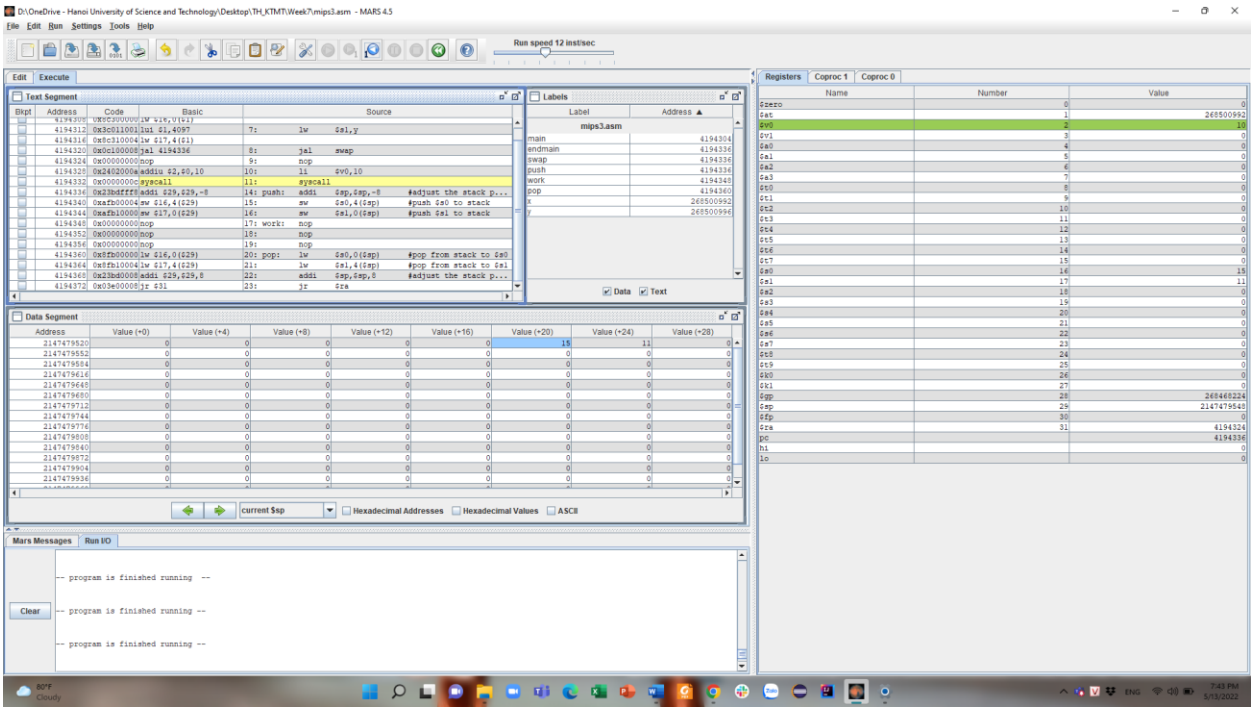
+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194336)



Sau khi kết thúc chương trình:

+ $ra ghi địa chỉ của câu lệnh liền sau câu lệnh jal là nop. ($ra = 4194324)

+ pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194336)



Exercise 4:

<u>a, Khi chương trình sử dụng thanh ghi $fp</u>

- *Chương trình:*

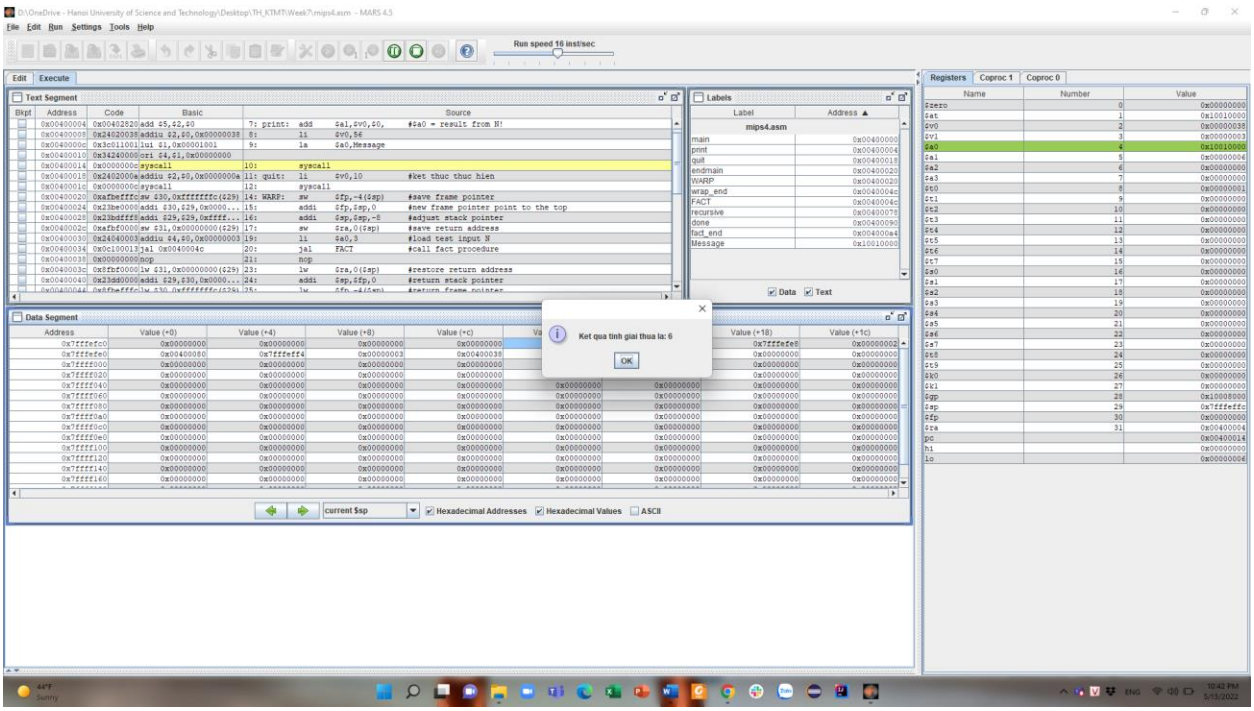
```
#Laboratory Exercise 7, Home Assignment 4
.data
Message:        .asciiz "Ket qua tinh giai thua la: "
.text
main:   jal     WARP

print:  add     $a1,$v0,$0,     #$a0 = result from N!
        li      $v0,56
        la      $a0,Message
        syscall
quit:   li      $v0,10          #ket thuc thuc hien
        syscall
endmain:
WARP:   sw      $fp,-4($sp)     #save frame pointer
        addi    $fp,$sp,0       #new frame pointer point to the top
        addi    $sp,$sp,-8      #adjust stack pointer
        sw      $ra,0($sp)      #save return address

        li      $a0,3           #load test input N
        jal     FACT            #call fact procedure
        nop

        lw      $ra,0($sp)      #restore return address
        addi    $sp,$fp,0       #return stack pointer
        lw      $fp,-4($sp)     #return frame pointer
        jr      $ra
wrap_end:
FACT:   sw      $fp,-4($sp)             #save frame pointer
        addi    $fp,$sp,0               #new frame pointer point to stack's top
        addi    $sp,$sp,-12             #allocate space for $fp,$ra,$a0 in stack
        sw      $ra,4($sp)              #save return address
        sw      $a0,0($sp)              #save $a0 register
        slti    $t0,$a0,2               #if input argument N < 2
        beq     $t0,$zero,recursive     #if it is false (($a0 = N) >=2)
        nop
        li      $v0,1                   #return the result N!=1
        j       done
        nop
recursive:      addi    $a0,$a0,-1      #adjust input argument
                jal     FACT            #recursive call
                nop
                lw      $v1,0($sp)      #load a0
                mult    $v1,$v0         #compute the result
                mflo    $v0
done:           lw      $ra,4($sp)      #restore return address
                lw      $a0,0($sp)      #restore a0
                addi    $sp,$fp,0       #restore stack pointer
                lw      $fp,-4($sp)     #restore frame pointer
                jr      $ra             #jump to calling
fact_end:
```

- *Kết quả:*



- *Giải thích:*

+ Trước khi vào chương trình con WARP:

$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400000

+ Trong hàm WARP và chạy xong sw

$sp	29	0x7fffeff4
$fp	30	0x7fffeffc
$ra	31	0x00400004
pc		0x00400030

Stack:

Value (+14)	Value (+18)	Value (+1c)
0x00400004	0x00000000	0x00000000

$ra(main) = 0x00400004

0($sp) = 0x00400004

$fp(main) = 0x00000000

0($fp) = 0x00000000

+ Trong hàm FACT ($a0=3) và chạy xong sw

$sp	29	0x7fffefe8
$fp	30	0x7fffeff4
$ra	31	0x00400038
pc		0x00400060

Stack:

Value (+8)	Value (+c)	Value (+10)	Value (+14)
0x00000003	0x00400038	0x7fffeffc	0x00400004

$a0=3

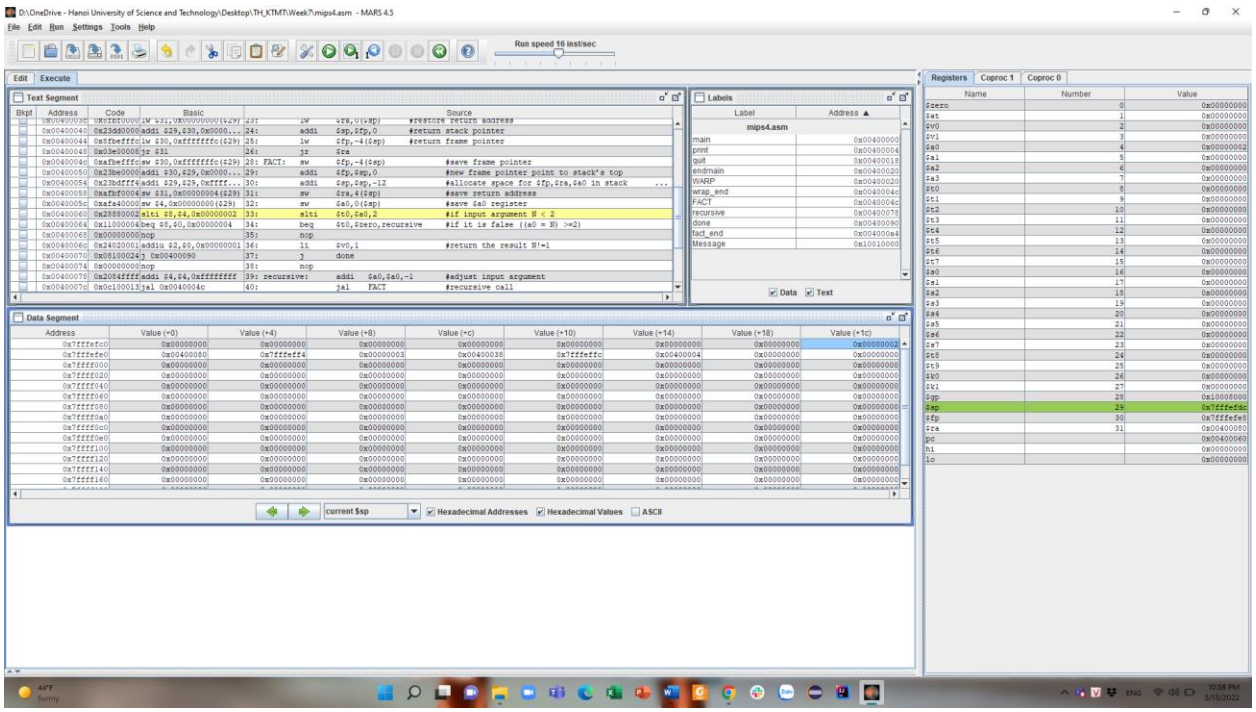
$ra(WARP)= 0x00400038

$fp(WARP)= 0x7fffeffc

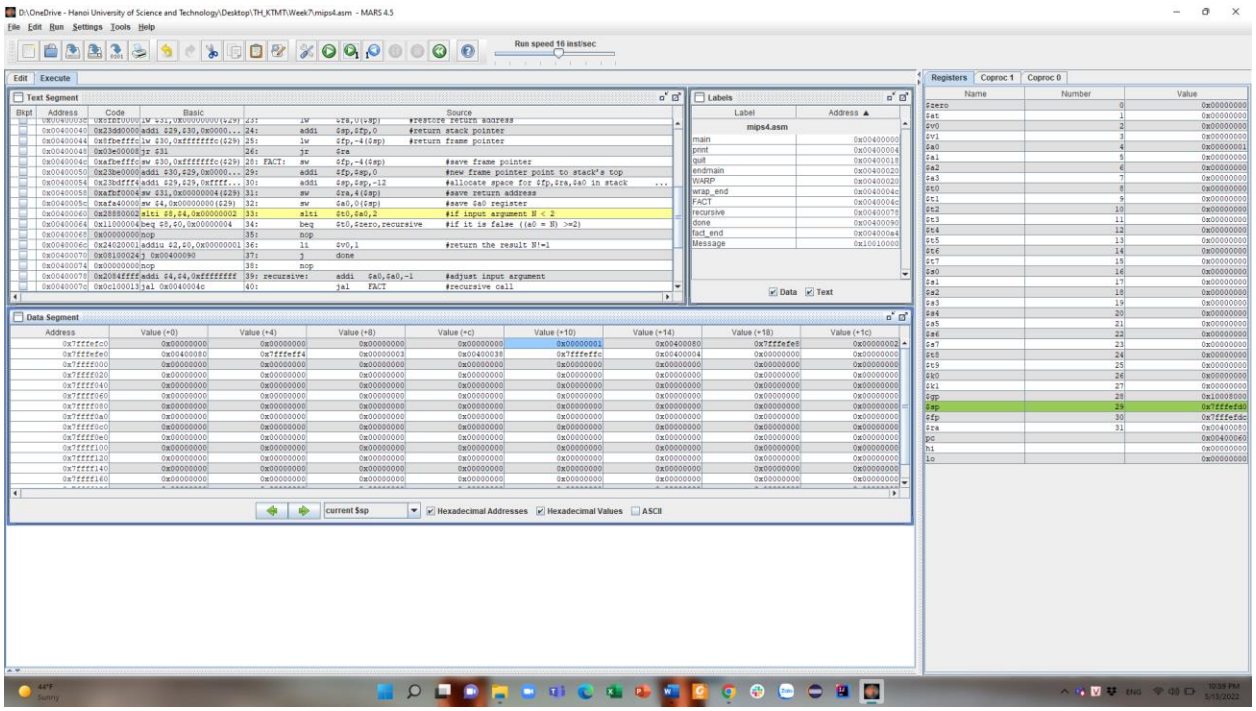
0($sp)= 0x00000003

0($fp) = 0x00400004

+ Trong hàm FACT($a0=2) và chạy xong sw



+ Trong hàm FACT($a0=1) và chạy xong sw



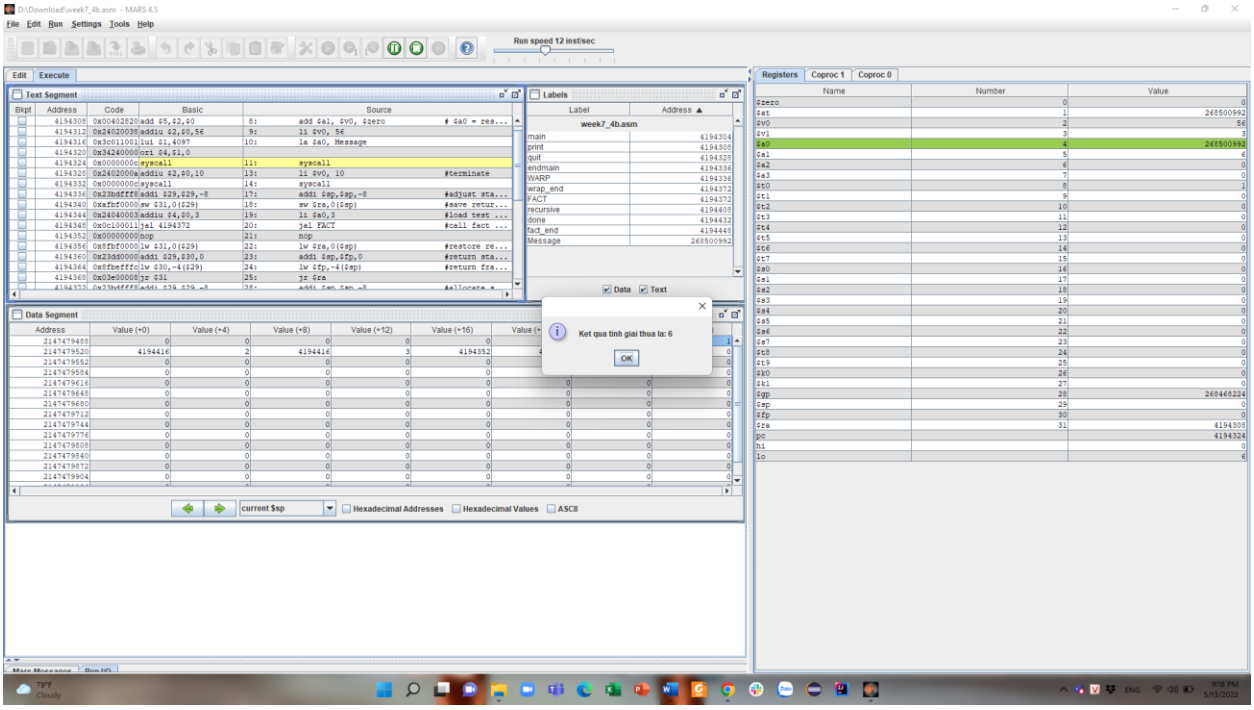
<u>b, Khi chương trình không sử dụng thanh ghi $fp:</u>

- *Chương trình:*

```
#Laboratory Exercise 7, Home Assignment 4
.data
Message: .asciiz "Ket qua tinh giai thua la: "
.text
main:
        jal WARP
print:
        add $a1, $v0, $zero             # $a0 = result from N!
        li $v0, 56
        la $a0, Message
        syscall
quit:
        li $v0, 10                      #terminate
        syscall
endmain:
WARP:
        addi $sp,$sp,-8                 #adjust stack pointer (3)
        sw $ra,0($sp)                   #save return address (4)
        li $a0,3                        #load test input N
        jal FACT                        #call fact procedure
        nop
        lw $ra,0($sp)                   #restore return address (5)
        addi $sp,$fp,0                  #return stack pointer (6)
        lw $fp,-4($sp)                  #return frame pointer (7)
        jr $ra
wrap_end:
FACT:
        addi $sp,$sp,-8                 #allocate space for $fp,$ra,$a0 in stack
        sw $ra,4($sp)                   #save return address
        sw $a0,0($sp)                   #save $a0 register

        slti $t0,$a0,2                  #if input argument N < 2
        beq $t0,$zero,recursive         #if it is false (($a0 = N) >=2)
        nop
        li $v0,1                        #return the result N!=1
        j done
        nop
recursive:
        addi $a0,$a0,-1                 #adjust input argument
        jal FACT                        #recursive call
        nop
        lw $v1,0($sp)                   #load a0
        mult $v1,$v0                    #compute the result
        mflo $v0
done:
        lw $ra,4($sp)                   #restore return address
        lw $a0,0($sp)                   #restore a0
        addi $sp,$sp,8                  #restore frame pointer
        jr $ra                          #jump to calling
fact_end:
```

- *Kết quả:*



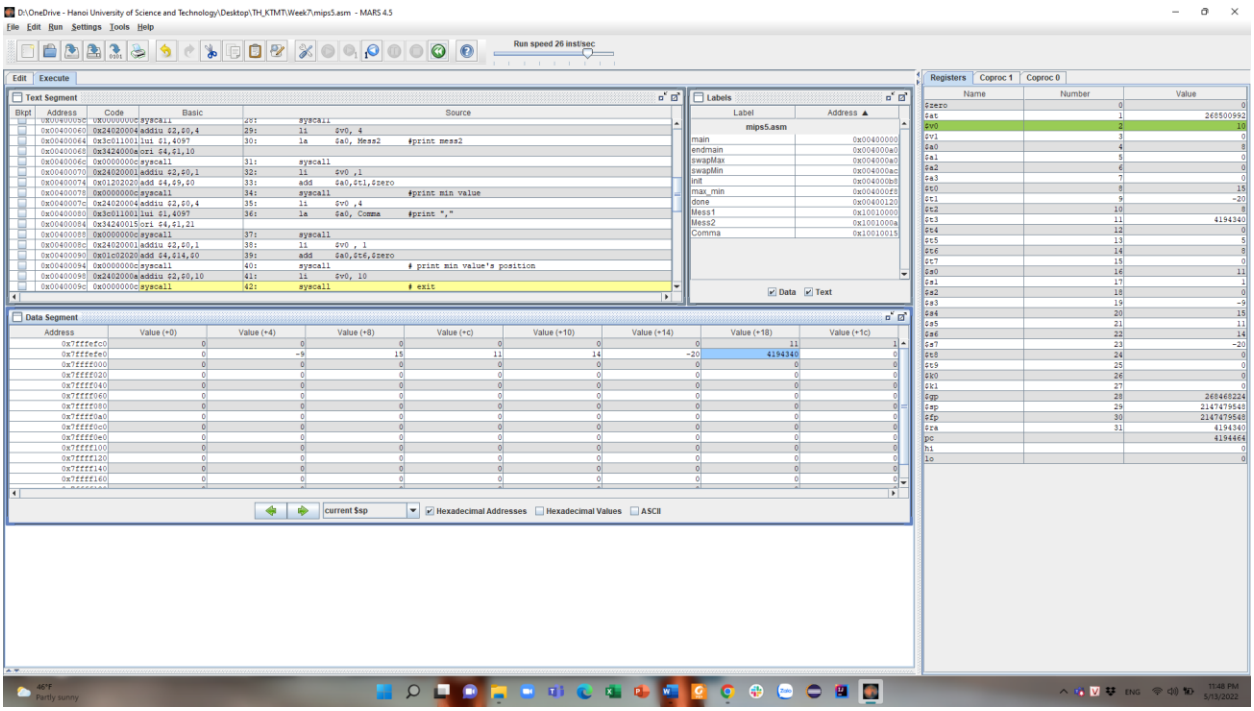
Exercise 5:

- *Chương trình:*

```
#Laboratory Exercise 7, Assignment 5
.data
Mess1:  .asciiz "Largest: "
Mess2:  .asciiz "\nSmallest:"
Comma:  .asciiz ","
.text
main:   li      $s0, 11            # Load input
        li      $s1, 1
        li      $s2, 0
        li      $s3, -9
        li      $s4, 15
        li      $s5, 11
        li      $s6, 14
        li      $s7, -20
        jal     init
        nop
        li      $v0, 4
        la      $a0, Mess1       #print mess1
        syscall
        li      $v0 , 1
        add     $a0,$t0,$zero
        syscall                  #print max value
        li      $v0, 4
        la      $a0, Comma       #print ","
        syscall
        li      $v0 , 1
        add     $a0,$t5,$zero    # print max value's position
        syscall
        li      $v0, 4
        la      $a0, Mess2       #print mess2
        syscall
        li      $v0 ,1
        add     $a0,$t1,$zero
        syscall                  #print min value
        li      $v0 ,4
        la      $a0, Comma       #print ","
        syscall
        li      $v0 , 1
        add     $a0,$t6,$zero
        syscall                  # print min value's position
        li      $v0, 10

        syscall                  # exit
endmain:
swapMax:        add     $t0,$t3,$zero   # set Max = $t3
                add     $t5,$t2,$zero   # set i of max = $t2
                jr      $ra
swapMin:        add     $t1,$t3,$zero   # set Min = $t3
                add     $t6,$t2,$zero   # set i of min = $t2
                jr      $ra
init:           add     $fp,$sp,$zero   # save address of origin sp
                addi    $sp,$sp, -36    # create space for stack
                sw      $s0, 0($sp)
                sw      $s1, 4($sp)
                sw      $s2, 8($sp)
                sw      $s3, 12($sp)
                sw      $s4, 16($sp)
                sw      $s5, 20($sp)
                sw      $s6, 24($sp)
                sw      $s7, 28($sp)
                sw      $ra, 32($sp)    #save $ra for main
                add     $t0,$s0,$zero   # set Max = $s0
                add     $t1,$s0,$zero   # set Min = $s0
                li      $t5, 0          # set $t5 to 0
                li      $t6, 0          # set $t6 to 0
                li      $t2, 0          # set $t2 to 0 , i = 0
max_min:        addi    $sp,$sp,4
                lw      $t3,-4($sp)
                sub     $t4, $sp, $fp   # check if meet $ra
                beq     $t4,$zero, done # if true then done
                addi    $t2,$t2,1       # i++
                sub     $t4,$t0,$t3     # Max - $t3
                bltzal  $t4, swapMax    #if Max - $t3 < 0, swap Max
                sub     $t4,$t3,$t1     # $t3 - Min
                bltzal  $t4, swapMin    # if $t3 â€“ Min < 0 , swap Min
                j       max_min         # repeat
done:           lw      $ra, -4($sp)    # load #$ra
                jr      $ra             # return
```

- *Kết quả:*



- *Giải thích:*
  Thanh ghi $t0 lưu giá trị lớn nhất: (t0) = 15
  Thanh ghi $t1 lưu giá trị nhỏ nhất: (t1) = -20