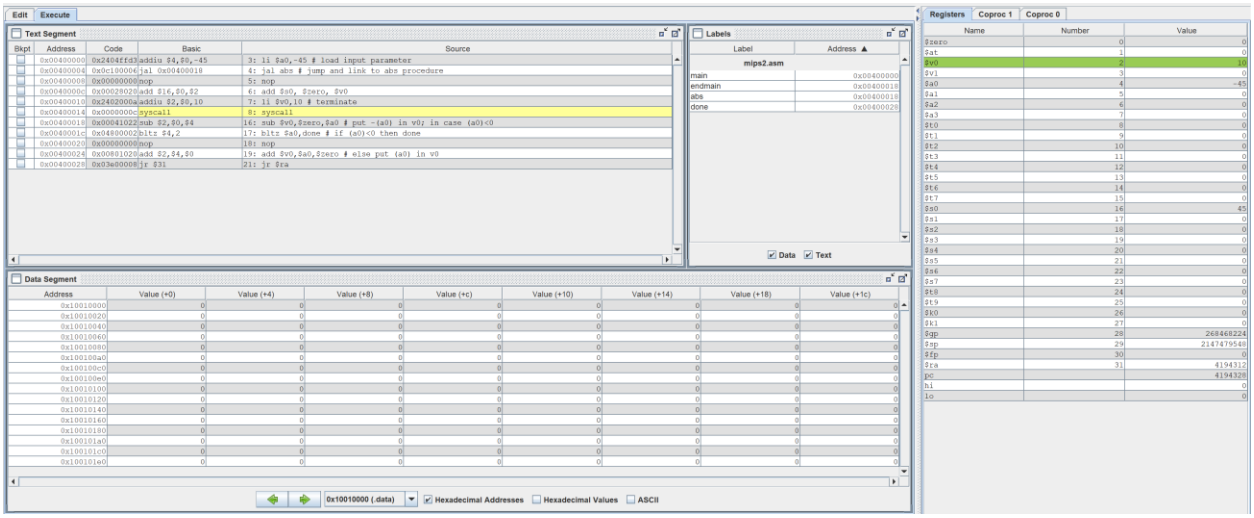
LABORATORY 7

Họ tên: Vũ Quốc Bảo
MSSV: 20225694

Assignment 1:

```
.text
main:
li $a0,-45 # load input parameter
jal abs # jump and link to abs procedure
nop
add $s0, $zero, $v0
li $v0,10 # terminate
syscall
endmain:
#---------------------------------------------------------------
# function abs
# param[in] $a1 the interger need to be gained the absolute value
# return $v0 absolute value
#---------------------------------------------------------------
abs:
sub $v0,$zero,$a0 # put -(a0) in v0; in case (a0)<0
bltz $a0,done # if (a0)<0 then done
nop
add $v0,$a0,$zero # else put (a0) in v0
done:
jr $ra
```

- Kết quả:



- Giải thích:

+ Trước khi nhảy đến chương trình con ***abs***:

- Chương trình khởi tạo giá trị cho thanh ghi $a0.
- $ra = 0.
- pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194304)

+ Sau khi nhảy đến chương trình con ***abs***:

- lệnh ***jal*** lưu địa chỉ trở về $ra. ($ra = 4194312)
- pc ghi địa chỉ của câu lệnh đang trỏ tới. (pc = 4194328)
- giá trị tuyệt đối của $a0 sẽ được lưu vào thanh ghi $v0.
- điều kiện ***bltz $a1, done*** kiểm tra xem giá trị đầu vào có âm hay không. Trong trường hợp này, vì giá trị là âm, nó nhảy tới ***done*** và kết thúc. Còn nếu không âm, giá trị đó sẽ được sao chép vào $v0.
- sau khi hoàn thành chương trình con ***abs***, chương trình quay trở lại lệnh sau ***jal*** là lệnh ***nop*** trong ***main***. Khi này thì giá trị tuyệt đối của $a0 đã được lưu vào trong $v0.

+ Nhảy ra khỏi ***abs***:

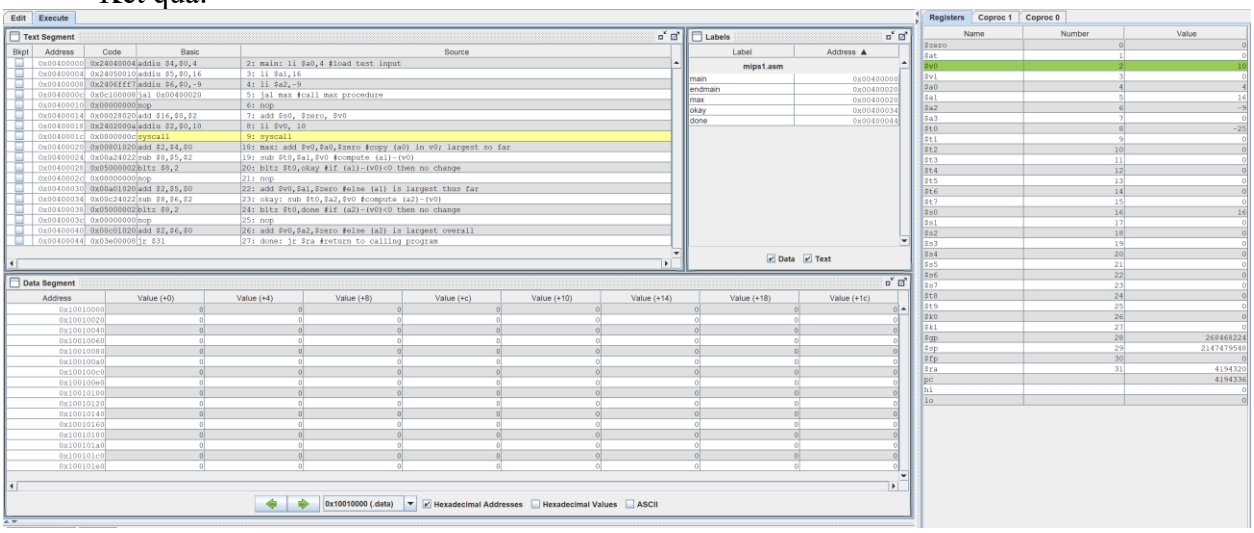
- $ra = 41943112.
- pc = 4194328.
- lưu giá trị từ $v0 chuyển sang $s0. Gán $v0 = 10 rồi sử dụng lệnh ***syscall*** để kết thúc chương trình.

Assignment 2:

```
.text
main: li $a0,4 #load test input
li $a1,16
li $a2,-9
jal max #call max procedure
nop
add $s0, $zero, $v0
li $v0,10 # terminate
syscall
endmain:
#------------------------------------------------------------------
#Procedure max: find the largest of three integers
#param[in] $a0 integers
#param[in] $a1 integers
#param[in] $a2 integers
#return $v0 the largest value
#------------------------------------------------------------------
max: add $v0,$a0,$zero #copy (a0) in v0; largest so far
sub $t0,$a1,$v0 #compute (a1)-(v0)
bltz $t0,okay #if (a1)-(v0)<0 then no change
nop
```

```
add $v0,$a1,$zero #else (a1) is largest thus far
okay: sub $t0,$a2,$v0 #compute (a2)-(v0)
bltz $t0,done #if (a2)-(v0)<0 then no change
nop
add $v0,$a2,$zero #else (a2) is largest overall
done: jr $ra #return to calling program
```

- Kết quả:



- Giải thích:

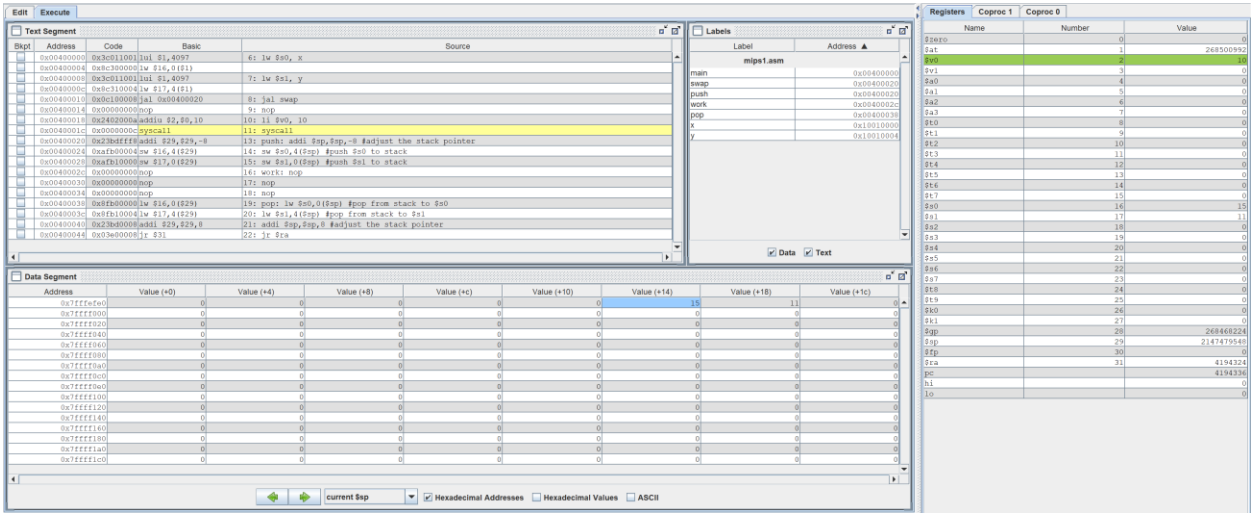
+ Khởi tạo giá trị cho 3 thanh ghi $a0, $a1, $a2 là 4, 16, -9.
+ $ra = 4194320.
+ Lệnh ***jal*** max gọi hàm ***max***, điều này làm chương trình nhảy đến địa chỉ của hàm ***max***.
+ Trong hàm ***max***:
    - giá trị $a0 sẽ được sao chép vào thanh ghi $v0, đây được coi là giá trị lớn nhất cho đến thời điểm này.
    - so sánh $a1 với $v0. Nếu $a1 < $v0 thì không có gì xảy ra. Còn ngược lại, $v0 sẽ được cập nhật thành giá trị của $a1.
    - tương tự với $a2 trong hàm ***okay***.
    - sau khi kết thúc, lệnh ***jr $ra*** sẽ đưa chương trình quay về lệnh sau lệnh ***jal***, tức là lệnh ***nop*** ở ***main***.
+ Ở đây, giá trị lớn nhất trong 3 thanh ghi đã được lưu vào $v0.
+ Chuyển giá trị từ $v0 sang $s0.
+ Sau đó, gán $v0 = 10 rồi sử dụng lệnh ***syscall*** để kết thúc chương trình.

<u>Assignment 3:</u>

```
.data
x: .word 11
y: .word 15
.text
main:
```

```
lw $s0, x
lw $s1, y
jal swap
nop
li $v0, 10
syscall
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:

+ Giá trị của biến x được nạp vào thanh ghi $s0 bằng lệnh ***lw $s0, x***.
+ Giá trị của biến y được nạp vào thanh ghi $s1 bằng lệnh ***lw $s1, y***.
+ Chương trình gọi hàm ***swap*** bằng lệnh ***jal swap***. Điều này làm chương trình nhảy đến địa chỉ của hàm ***swap***.
+ Trong hàm ***swap***:

- Đầu tiên, một không gian trên ngăn xếp được cấp phát bằng cách giảm con trỏ ngăn xếp $sp đi 8 byte bằng lệnh ***addi $sp, $sp, -8***.
- Các giá trị của $s0 và $s1 được đẩy vào ngăn xếp. $s0 được đẩy trước và lưu tại địa chỉ 4($sp), sau đó $s1 được đẩy và lưu tại địa chỉ 0($sp).

- Sau khi giá trị đã được đẩy vào ngăn xếp, các lệnh ***nop*** được sử dụng làm lùi bước để có thời gian cho việc thực hiện đẩy và lấy giá trị trên ngăn xếp.
- Các giá trị đó sẽ được lấy từ ngăn xếp và sao chép vào các thanh ghi $s0 và $s1.
- Cuối cùng, không gian trên ngăn xếp được giải phóng bằng cách tăng con trỏ ngăn xếp $sp lên 8 byte bằng lệnh ***addi $sp, $sp, 8***, và hàm kết thúc bằng lệnh ***jr $ra*** để quay lại lệnh sau lệnh gọi hàm.

+ Chương trình tiếp tục thực hiện lệnh ***nop*** sau lệnh ***jal***.
+ Sau đó, gán $v0 = 10 rồi sử dụng lệnh ***syscall*** để kết thúc chương trình.

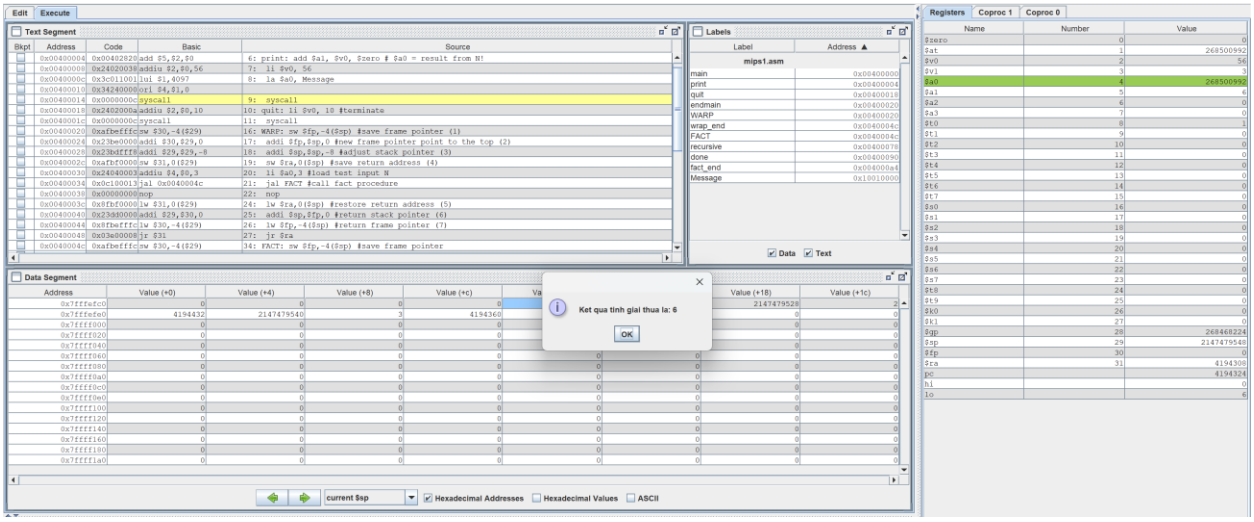
→ Vì vậy, trong quá trình này, giá trị của biến x và y (hay giá trị của 2 thanh ghi $s0 và $s1) đã được hoán đổi bằng cách sử dụng một hàm ***swap***.

Assignment 4:

```
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:
#---------------------------------------------------------------------
#Procedure WARP: assign value and call FACT
#---------------------------------------------------------------------
WARP: sw $fp,-4($sp) #save frame pointer (1)
 addi $fp,$sp,0 #new frame pointer point to the top (2)
 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:
#---------------------------------------------------------------------
#Procedure FACT: compute N!
#param[in] $a0 integer N
#return $v0 the largest value
#---------------------------------------------------------------------
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:



At the begin of WRAP, $sp = 0x7fffeffec

$ra (4)	7fffeff4 ← new $sp (3) addi $sp,$sp,-8
$fp	7fffeff8 (1) sw $fp,-4($sp)
	7fffeffc ← new $fp (2) addi $fp,$sp,0

At the end of WRAP, $sp = 0x7fffeff4

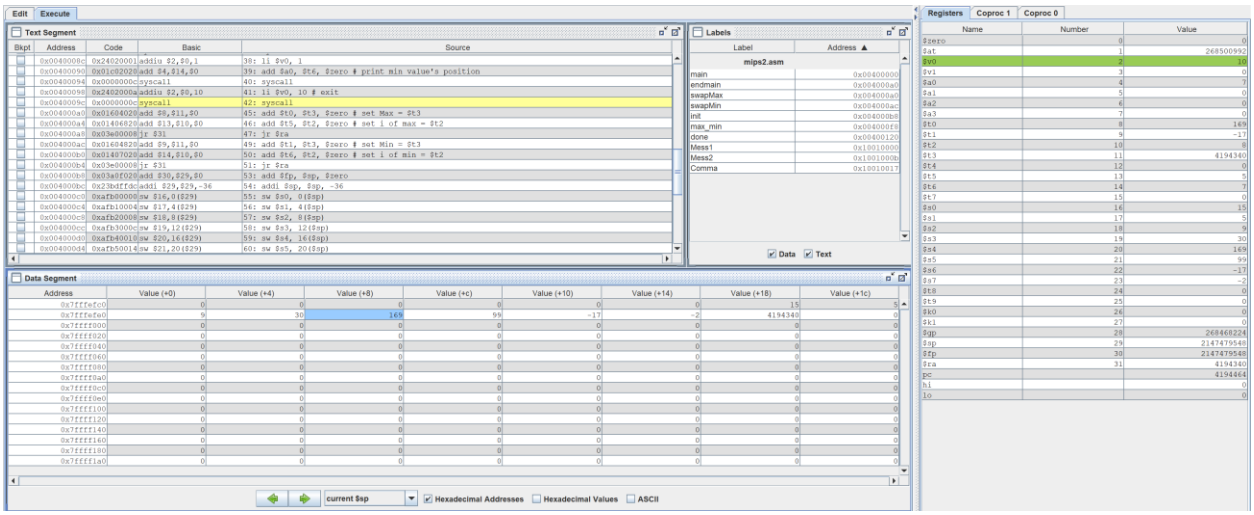
$a0	1
$ra	0x00400080
$fp	0x7fffefe8
$a0	2
$ra	0x00400080
$fp	0x7fffeff4
$a0	3
$ra	0x00400038
$fp	0x7fffeffc
$ra (4)	7fffeff4 → restore $ra (5)
$fp	7fffeff8 → restore $fp (7)
	7fffeffc → restore $sq (6)

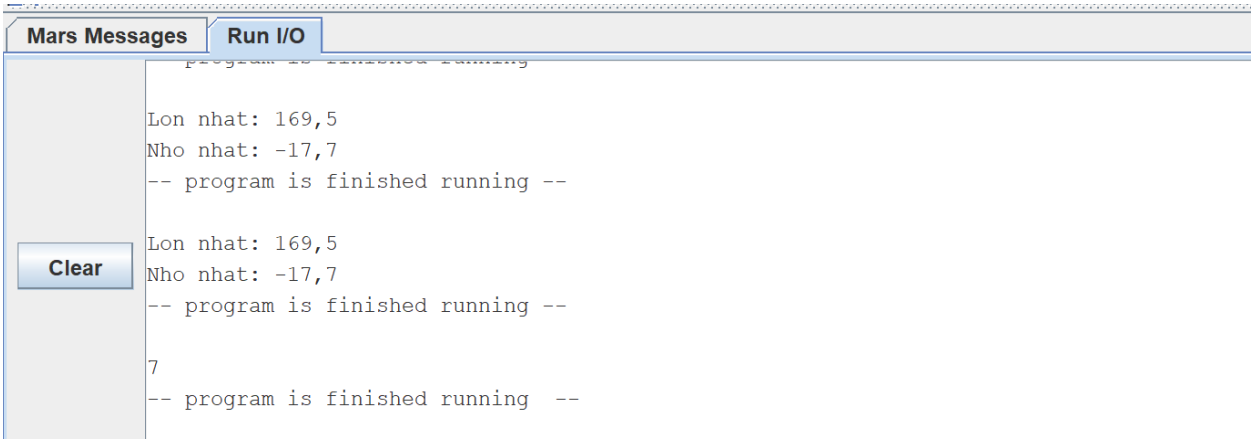
Assignment 5:

```
.data
Mess1: .asciiz "Lon nhat: "
Mess2: .asciiz "\nNho nhat: "
Comma: .asciiz ","
.text
main: # khởi tạo giá trị cho thanh ghi $s0-$s7
li $s0, 15
li $s1, 5
li $s2, 9
li $s3, 30
li $s4, 169
li $s5, 99
li $s6, -17
li $s7, -2
jal init
```

```
nop
li $v0, 4
la $a0, Mess1 # print Mess1
syscall
li $v0, 1
add $a0, $t0, $zero # print max value
syscall
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 # print min value
syscall
li $v0, 4
la $a0, Comma # print ","
syscall
li $v0, 1
add $a0, $t6, $zero # print min value's position
syscall
li $v0, 10 # exit
syscall
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
addi $sp, $sp, -36
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)
add $t0, $s0, $zero # set Max = $s0
# $t0 lưu giá trị max
add $t1, $s0, $zero # set Min = $s0
# $t1 lưu giá trị min
li $t5, 0
li $t6, 0
li $t2, 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
addi $t2, $t2, 1 # i++
sub $t4, $t0, $t3
bltzal $t4, swapMax
sub $t4, $t3, $t1
bltzal $t4, swapMin
j max_min # repeat
done:
lw $ra, -4($sp)
jr $ra
```




Mars Messages
Run I/O
Clear
Lon nhat: 169,5
Nho nhat: -17,7
-- program is finished running --
Lon nhat: 169,5
Nho nhat: -17,7
-- program is finished running --
7
-- program is finished running --