THỰC HÀNH KIẾN TRÚC MÁY TÍNH

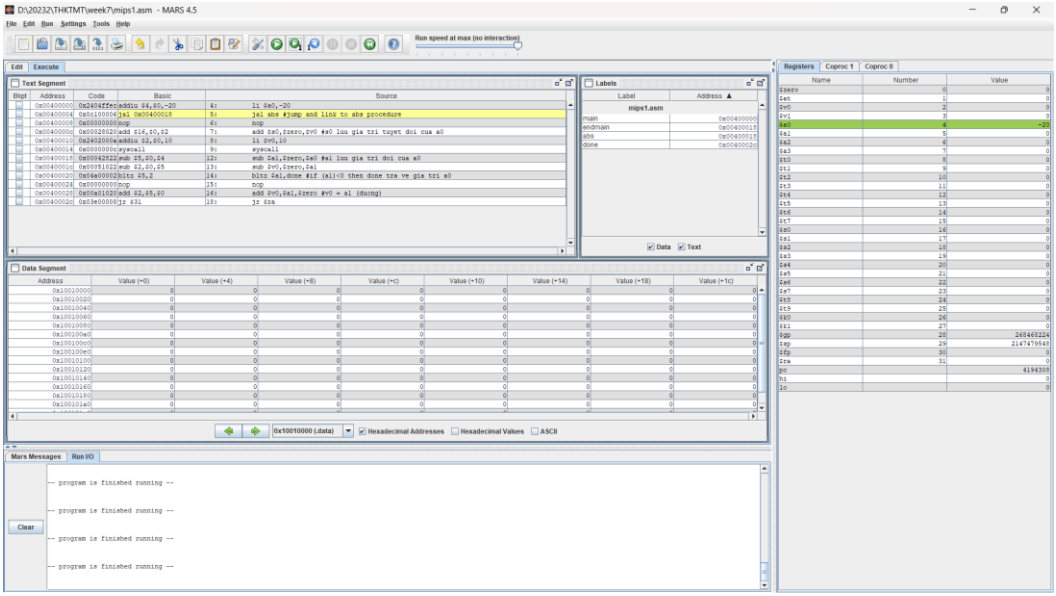
WEEK 7

Bùi Quang Hưng – 20225849

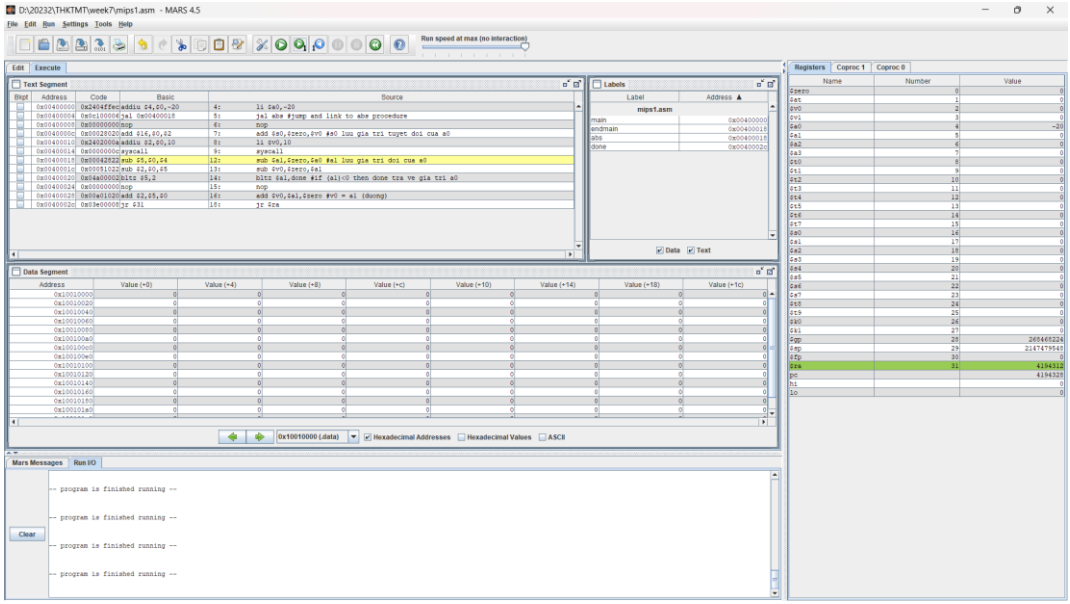
Assignment 1

```
#Laboratory Exercise 7, Assignment 1
.text
main:
        li $a0,-2206
        jal abs #jump and link to abs procedure
        nop
        add $s0,$zero,$v0 #s0 luu gia tri tuyet doi cua a0
        li $v0,10
        syscall
endmain:
abs:
        sub $a1,$zero,$a0 #a1 luu gia tri doi cua a0
        sub $v0,$zero,$a1
        bltz $a1,done #if (a1)<0 then done tra ve gia tri a0
        nop
        add $v0,$a1,$zero #v0 = a1 (duong)
done:
        jr $ra
        #Gia tri ra =pc+4(tai thoi diem truoc khi thuc hien jal), gia tri pc=ra+16
```

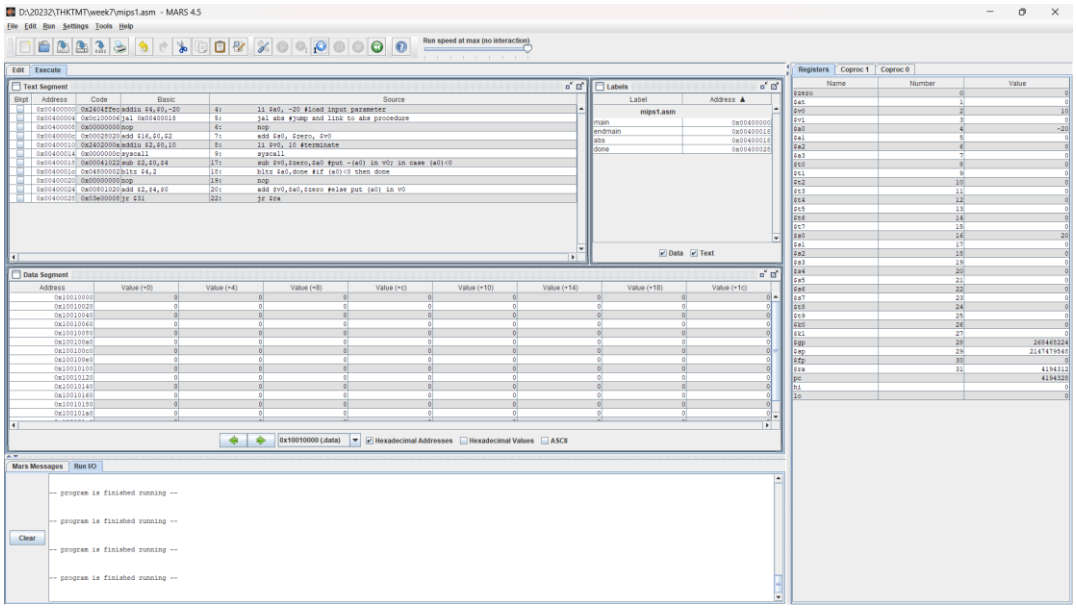
- Trước khi chạy lệnh jal abs:



- Sau khi chạy lệnh jal abs:



- Kết quả sau khi chạy xong chương trình:



- Nhận xét:

+ Khi chạy lệnh jal abs (địa chỉ lệnh 0x00400004) thì thanh ghi $ra được gán bằng địa chỉ của câu lệnh tiếp theo (0x00400008) và thanh ghi pc được gán bằng địa chỉ 0x00400018 (địa chỉ tại nhãn abs).

+ Kết quả cuối cùng ta lấy được giá trị tuyệt đối của số được nạp vào trong thanh ghi $a0 và ghi kết quả đó vào thanh ghi $s0.

$a0	4	-20
$s0	16	20

Assignment 2

a.Code:

#Laboratory Exercise 7, Home Assignment 2

.text

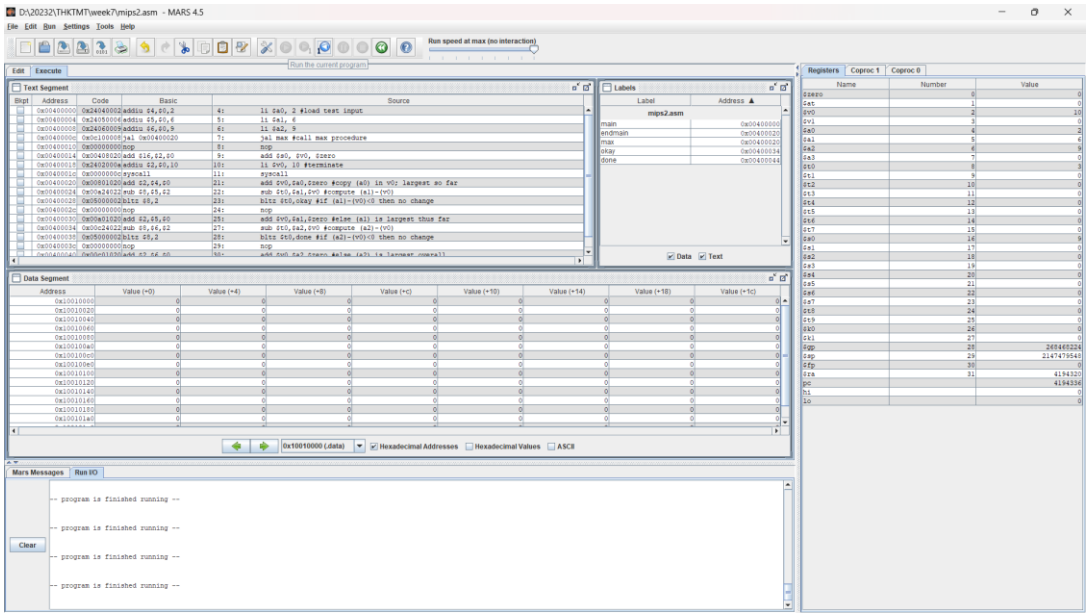
main:

```
    li $a0, 2 #load test input
    li $a1, 6
    li $a2, 9
    jal max #call max procedure
    nop
    add $s0, $v0, $zero
    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
```

b.Nhận xét

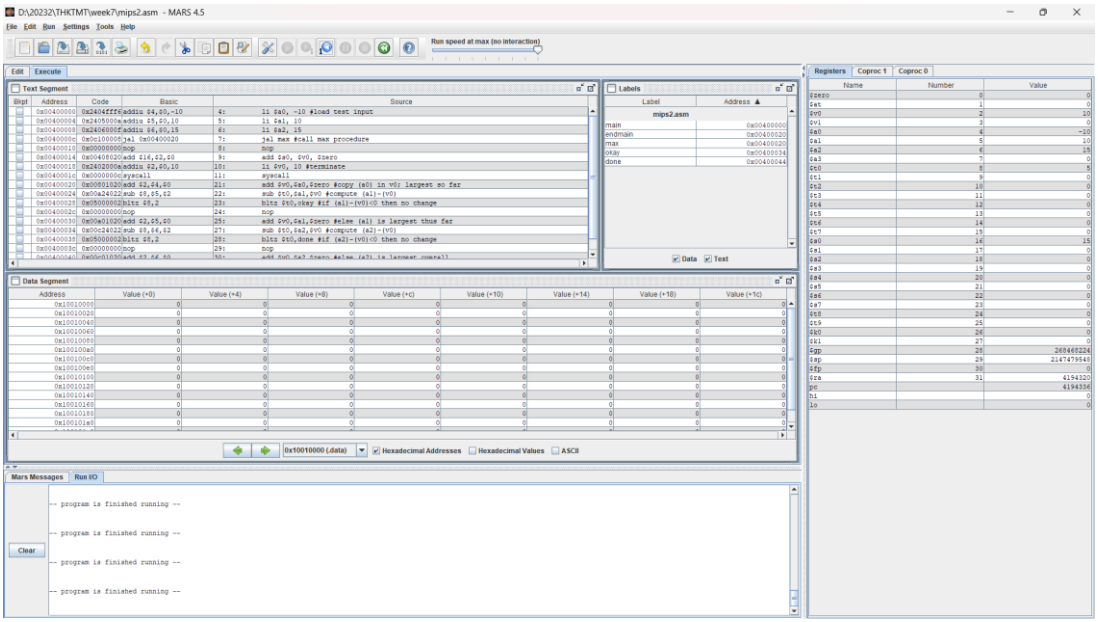
- Kết quả chạy chương trình mẫu:



+ Ta thấy thanh ghi $s0 chứa giá trị lớn nhất: $s0 = 9

$s0	16	9

- Sau khi thay các giá trị thanh ghi: $a0 =-10, $a1 = 10, $a2 = 15



- Kết quả thu được giá trị lớn nhất $s0 = 15

$s0	16	15

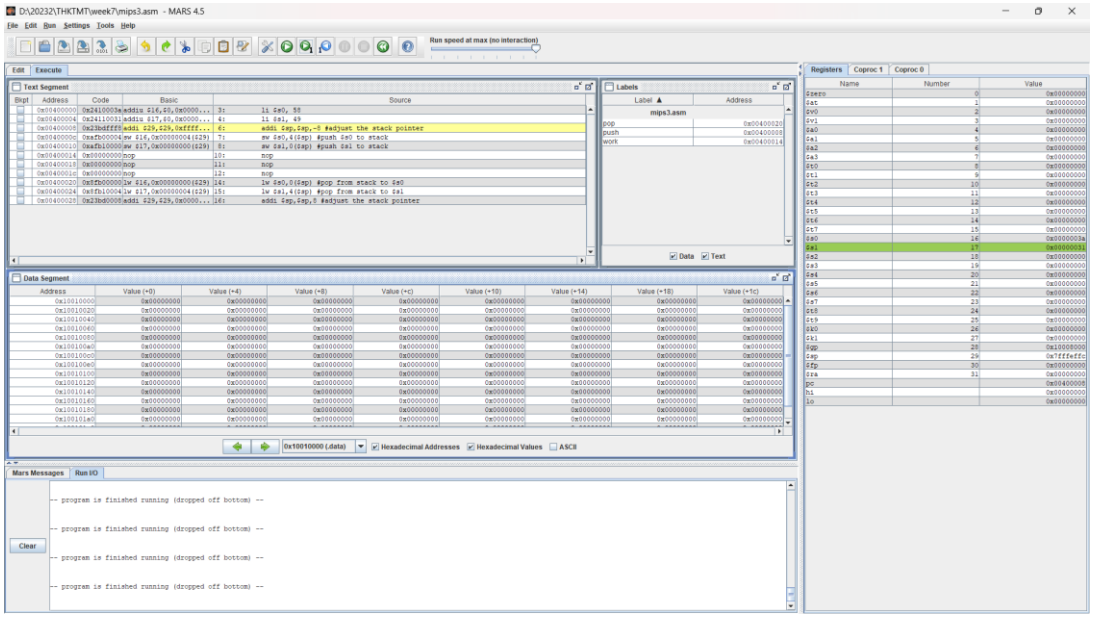
- Khi chạy lệnh jal thì thanh ghi $ra được gán bằng giá trị của địa chỉ của câu lệnh tiếp theo sau jal trong nhãn main. Thanh ghi pc được gán bằng địa chỉ của nhãn max để câu lệnh tiếp tục được thực hiện bắt đầu từ nhãn max. Sau khi chạy đến jr $ra thì pc được gán bằng địa chỉ trong $ra (địa chỉ của nop).

Assignment 3

a.Code:

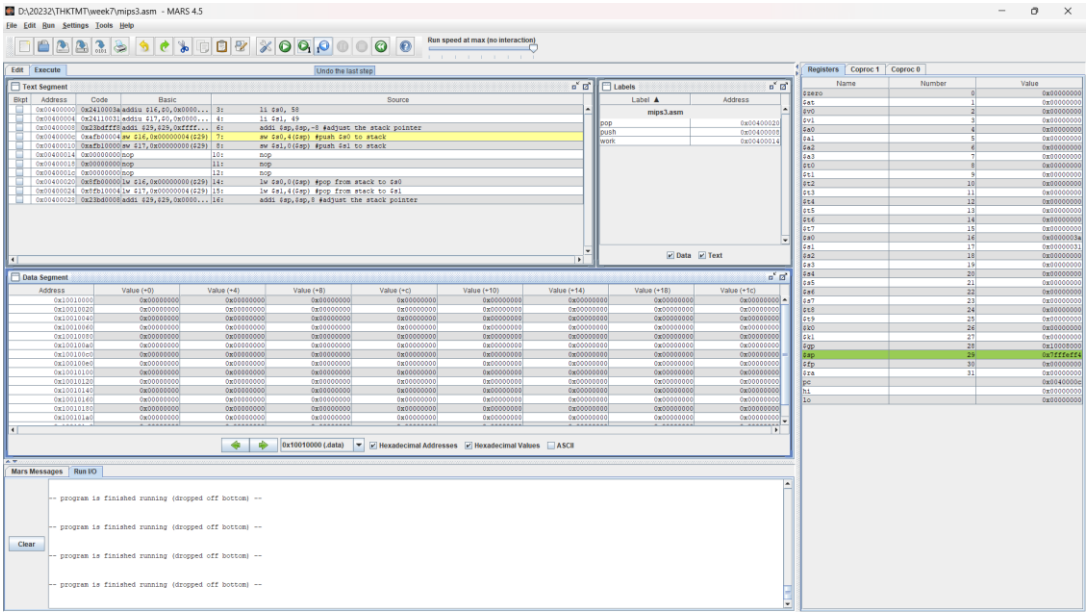
```
#Laboratory Exercise 7, Home Assignment 3
.text
        li $s0, 58
        li $s1, 49
push:
        addi $sp,$sp,-8 #adjust the stack pointer
        sw $s0,4($sp) #push $s0 to stack
        sw $s1,0($sp) #push $s1 to stack
work:
        nop
        nop
        nop
pop:
        lw $s0,0($sp) #pop from stack to $s0
        lw $s1,4($sp) #pop from stack to $s1
        addi $sp,$sp,8 #adjust the stack pointer
```

- Trước khi chạy lệnh addi ở nhãn push:



+ Ta thấy $sp = 0x7fffeffc

- Sau khi chạy lệnh addi ở nhãn push:

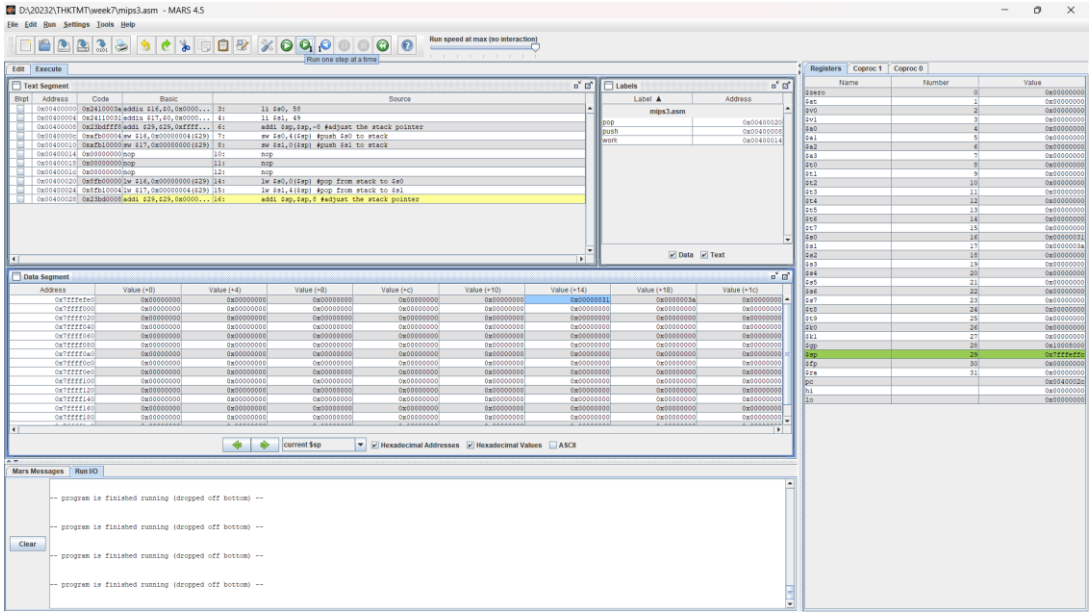


+ Ta thấy lúc này thanh ghi $sp = 0x7fffeff4

- Thanh ghi $sp được lùi đi 8 bytes, tức là có sự cấp phát bộ nhớ cho stack 8 bytes.

- Tiếp theo chương trình lần lượt ghi các giá trị của $s0 vào $sp+4 và $s1 vào giá trị của $s1 vào $sp+0.

- Sau khi thực hiện lệnh **addi** ở nhãn **pop:**



+ Thực hiện đổi chỗ hai số bằng cách load giá trị tại địa chỉ $sp + 0 vào $s0, load giá trị tại địa chỉ $sp + 4 vào $s1

+ Lệnh add $sp,$sp,8 để di chuyển con trỏ $sp về lại vị trí ban đầu 0x7fffeff4 + 8 = 0x7fffeffc

Assignment 4

a.Code:

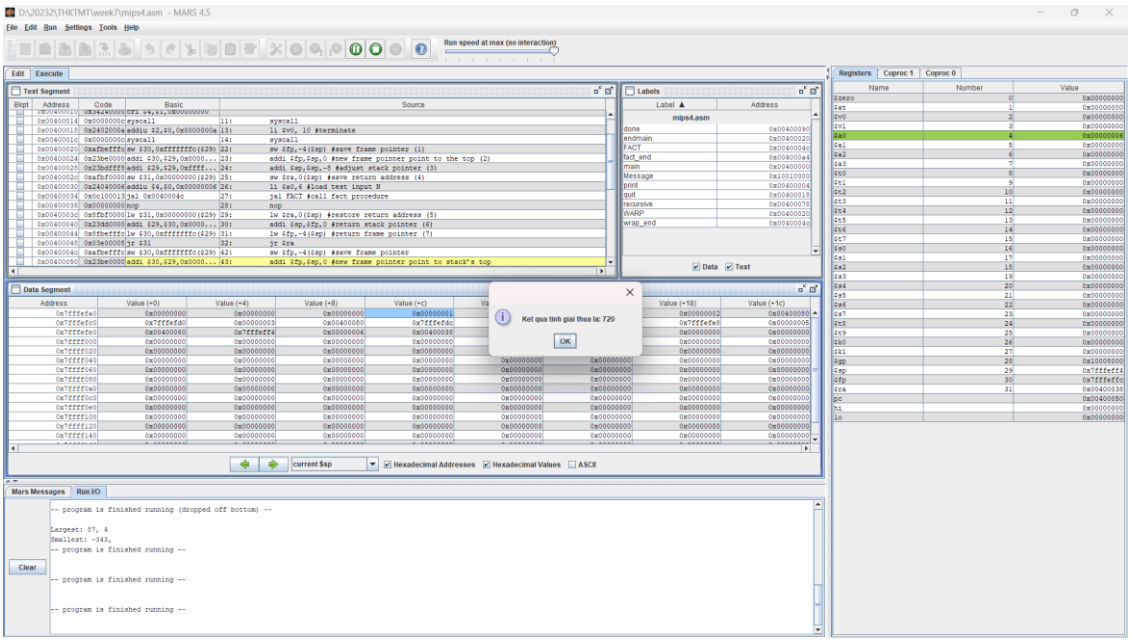
```
#Laboratory Exercise 7, Home Assignment 4
.data
Message: .asciiz "Ket qua tinh giai thua la: "
.text
main
	jal WARP
print:
	add $a1, $v0, $zero # $a0 = result from N!
	li $v0, 56
	la $a0, Message
	syscall
quit:
	li $v0, 10 #terminate
	syscall
endmain:
#---------------------------------------------------------------

#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:
```

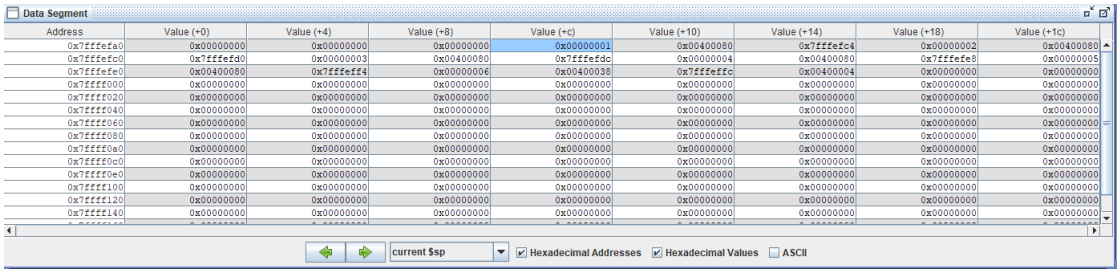
b. Nhận xét:

- Với n = 6, tức là ta có $a0 = 6



- Kết quả của chương trình 6! = 720. Kết quả đúng với lí thuyết.

- Sự thay đổi của thanh ghi $sp:



- Với trường hợp n=3 ta có bảng ngăn xếp sau:

$sp	
0x7fffefd0	$a0 = 0x00000001
0x7fffefd4	$ra = 0x00400080
0x7fffefd8	$fp = 0x7fffefe8
0x7fffefdc	$a0 = 0x00000002
0x7fffefe0	$ra = 0x00400080
0x7fffefe4	$fp = 0x7fffeff4
0x7fffefe8	$a0 = 0x00000003
0x7fffefec	$ra = 0x00400038
0x7fffeff0	$fp = 0x7fffeffc
0x7fffeff4	$ra = 0x00400004
0x7fffeff8	$fp = 0x00000000

Assignment 5

a. **Code:**

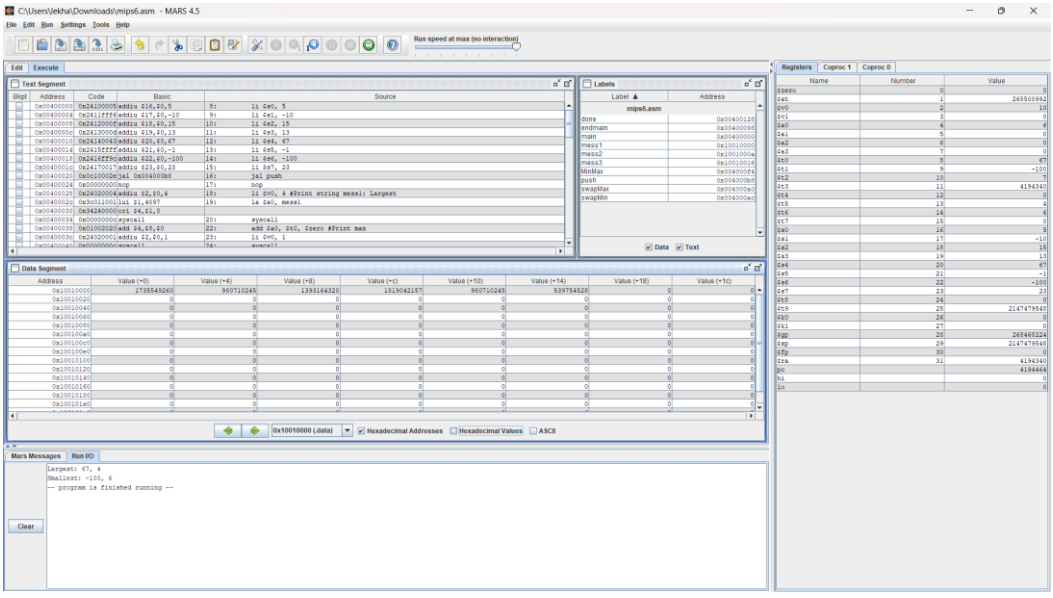
```
# Laboratory Exercise 7, Assignment 5
.data
mess1: .asciiz "Largest: "
mess2: .asciiz "\nSmallest: "
mess3: .asciiz ", "
.text
main:
        li $s0, 5
        li $s1, -10
        li $s2, 15
        li $s3, 13
        li $s4, 67
        li $s5, -1
        li $s6, -100
        li $s7, 23
        jal push
        nop
        li $v0, 4 #Print string mess1: Largest
        la $a0, mess1
        syscall

        add $a0, $t0, $zero #Print max
        li $v0, 1
        syscall

        li $v0, 4 #Print dấu phẩy
        la $a0, mess3
        syscall
```

```
        add $a0, $t5, $zero
        li $v0, 1 #Print register contain Max
        syscall

        li $v0, 4 # Print string mess2: Smallest
        la $a0, mess2
        syscall

        add $a0, $t1, $zero #Print Min
        li $v0, 1
        syscall

        li $v0, 4 # Print dấu phẩy
        la $a0, mess3
        syscall

        add $a0, $t6, $zero
        li $v0, 1 #Print register contain Min
        syscall
endmain:
        li $v0, 10 #Exit
        syscall
# $t0 = Max
# $t1 = Min
# $t5 = Index of Max
# $t6 = Index of Min
# $v0 : the largest of value
swapMax:
        add $t0,$t3,$zero
        add $t5,$t2,$zero
        jr $ra
swapMin:
        add $t1,$t3,$zero
        add $t6,$t2,$zero
        jr $ra
push:
        add $t9,$sp,$zero # Save address of origin $sp
        addi $sp,$sp, -32
        sw $s1, 0($sp)
        sw $s2, 4($sp)
        sw $s3, 8($sp)
        sw $s4, 12($sp)
        sw $s5, 16($sp)
```

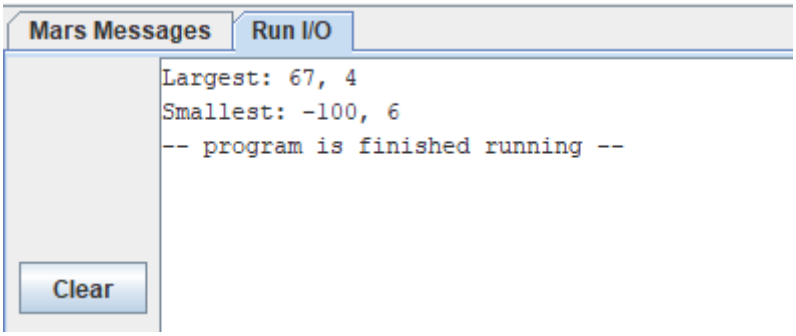
```
        sw $s6, 20($sp)
        sw $s7, 24($sp)
        sw $ra, 28($sp) # Save $ra for main
        add $t0,$s0,$zero # Max = $s0
        add $t1,$s0,$zero # Min = $s0
        li $t5, 0 # Index of Max to 0
        li $t6, 0 # Index of Min to 0
        li $t2, 0 # i = 0
MinMax:
        addi $sp, $sp, 4
        lw $t3, -4($sp)
        sub $t4, $sp, $t9
        beq $t4, $zero, done # If $sp = $fp branch to the 'done'
        nop
        addi $t2, $t2, 1 # index++
        sub $t4, $t0, $t3
        bltzal $t4, swapMax # If $t3 > Max branch to the swapMax
        nop
        sub $t4,$t3,$t1
        bltzal $t4, swapMin # If $t3 < Min branch to the swapMin
        nop
        j MinMax # Repeat
done:
        lw $ra, -4($sp)
        jr $ra # Return to calling program
```

b. Nhận xét

- Sau khi chạy xong chương trình trên:



- Kết quả màn hình Run I/O:


Mars Messages
Run I/O
Largest: 67, 4
Smallest: -100, 6
-- program is finished running --
Clear