**BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH**

**Assignment 1:** Tính giá trị tuyệt đối

**Source Code:**

#Laboratory Exercise 7 Home Assignment 1

.text

main: li $a0,-42 # 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] $a0 the integer 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:**

Sau khi thực hiện câu lệnh jal abs, chương trình sẽ nhảy xuống chương trình con abs

Lúc này thanh ghi ra có giá trị là địa chỉ của lệnh nop (câu lệnh ngay sau lệnh nhảy jal )

Do $a0 < 0 nên chương trình nhảy đến done quay về main để gán giá trị tuyệt đối vào $s0 với giá trị 42

**Assignment 2:** Tìm số lớn nhất trong 3 số

**Source Code:**

#Laboratory Exercise 7, Home Assignment 2

.text

main: li $a0,1 # load test input

li $a1,3

li $a2,5

jal max # call max procedure

nop

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

ok: 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:**

Khi chạy đến câu lệnh bltz trong chương trình max, do $a1 = 3 lớn hơn $v0 = 1 nên thực hiện tiếp lệnh gán lại giá trị cho $v0 (max), sau đó mới chuyển tới chương trình con ok để thực hiện tiếp so sánh $v0 với $a2.

Sau đó thực hiện lệnh jal để nhảy về lệnh nop trong main.

**Assignment 3**

**Source Code:**

#Laboratory Exercise 7, Home Assignment 3

.text

init: li $s0,3

li $s1,1

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

**Kết quả:**

Khi chạy hàm push, 2 giá trị của $s0, $s1 được push vào trong stack.

Khi chạy hàm pop, 2 giá trị của $s0, $s1 được pop ra ngược với thứ tự push.

=> Giá trị của 2 thanh ghi được hoán đổi.

Giá trị ban đầu: $s0 = 3, $s1 = 1

Giá trị sau khi chạy chương trình: $s0 = 1, $s1 = 3

**Assignment 4**

**Source Code:**

#Laboratory Exercise 7 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 $a0,6 # 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 stackâ€™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

jalFACT # 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:**

Với input n = 5: - > kết quả = 5! = 120

Sự thay đổi các thanh ghi:

Thanh ghi $sp:

⦁ Khi n = 3

Bảng thể hiện giá trị ngăn xếp:

$fp = 0x7fffefcd

$a0 = 0x0000001

$ra = 0x0040080

$fp = 0x7fffefe8

$a0 = 0x0000002

$ra = 0x0040080

$fp = 0x7fffeff4

$a0 = 0x0000003

$ra = 0x0040038

$fp = 0x7fffeffc

$ra = 0x0040004

**Assignment 5**

**Source Code:**

#Laboratory Exercise 7, Assignment 5

.data

MesOfMax: .asciiz "The largest value is "

MesOfMin: .asciiz "The smallest value is "

MesMaxIndex: .asciiz "The largest element is stored in $s"

MesMinIndex: .asciiz "The smallest element is stored in $s"

.text

main: li $s0, -1

li $s1, 0

li $s2, 1

li $s3, 2

li $s4, 3

li $s5, 4

li $s6, 5

li $s7, 6

jal init # call max procedure

nop

li $v0, 56

la $a0, MesMaxIndex # print index of max element

add $a1,$t8,$zero

syscall

la $a0, MesOfMax # print max value

add $a1,$t0,$zero

syscall

la $a0, MesMinIndex # print index of min value

add $a1,$t9,$zero

syscall

la $a0, MesOfMin # print min value

add $a1,$t1,$zero

syscall

li $v0, 10 # exit program

syscall

endmain:

swapMax:add $t0,$t3,$zero # set Max = $t3

add $t8,$t2,$zero # set index of Max = $t2

jr $ra

swapMin:add $t1,$t3,$zero # set Min = $t3

add $t9,$t2,$zero # set index of Min = #$t2

jr $ra

init: add $fp,$sp,$zero # save address of origin sp

addi $sp,$sp, -32 # create space for stack

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 # set Max = $s0

add $t1,$s0,$zero # set Min = $s0

li $t8, 0 # set index of Max to 0

li $t9, 0 # set index of Min to 0

li $t2, 0 # set current index to 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, done

addi $t2,$t2,1 # increase index

sub $t4,$t0,$t3 # cal Max - $t3

bltzal $t4, swapMax # if Max < $t3, swap Max

sub $t4,$t3,$t1 # cal $t3 - Min

bltzal $t4, swapMin # if $t3 < Min, swap Min

j max\_min # repeat

done: lw $ra, -4($sp) # load #$ra

jr $ra # return to calling program

**Kết quả:**

Ta sử dụng các chương trình con để thực hiện assignment này.

+ swapMax: gán giá trị max là giá trị đang xét hiện tại, lưu lại index của nó.

+ swapMin: gán giá trị min là giá trị đang xét hiện tại, lưu lại index của nó.

+ init: hàm khởi tạo giá trị cho array cần xử lí, lưu chúng vào stack, chứa hàm con là max\_min (gọi hàm tìm giá trị max, min)

+ max\_min: Tìm giá trị max, min trong stack