**Bài tập thực hành week7**

1. **Assignment 1**

**Mã nguồn**

#Laboratory Exercise 7 Home 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,$a1 #put -(a0) in v0; in case (a0)<0

bltz $a1,done #if (a0)<0 then done

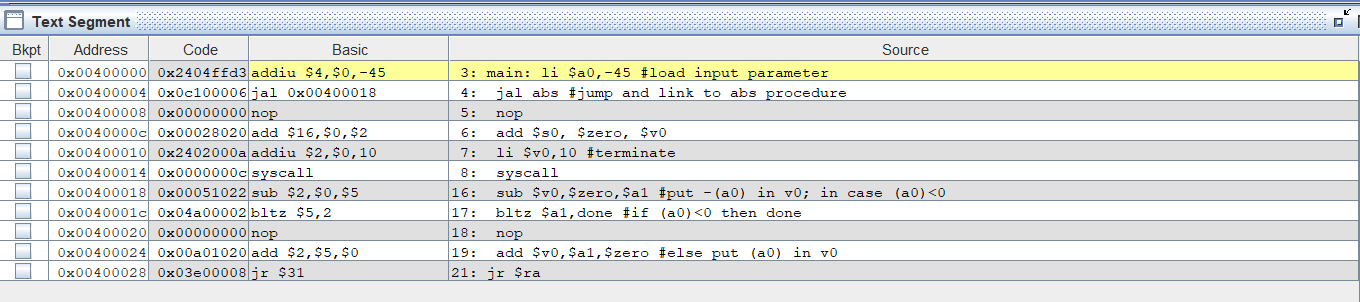
nop

add $v0,$a1,$zero #else put (a0) in v0

done:

jr $ra

**Kết quả :**













Ảnh có chứa bàn

Mô tả được tạo tự động

1. **Assignment 2**

**Mã nguồn**

#Laboratory Exercise 7, Home Assignment 2

.text

main: li $a0,8 #load test input

li $a1,6

li $a2,16

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

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 :**

Nạp phần tử :









Kết quả cuối cùng :Tìm ra số lớn nhất max = 16

Ảnh có chứa bàn

Mô tả được tạo tự động

1. **Assignment 3**

**Mã nguồn**

#Laboratory Exercise 7, Home Assignment 3

.text

li $s0 ,12

li $s1,4

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

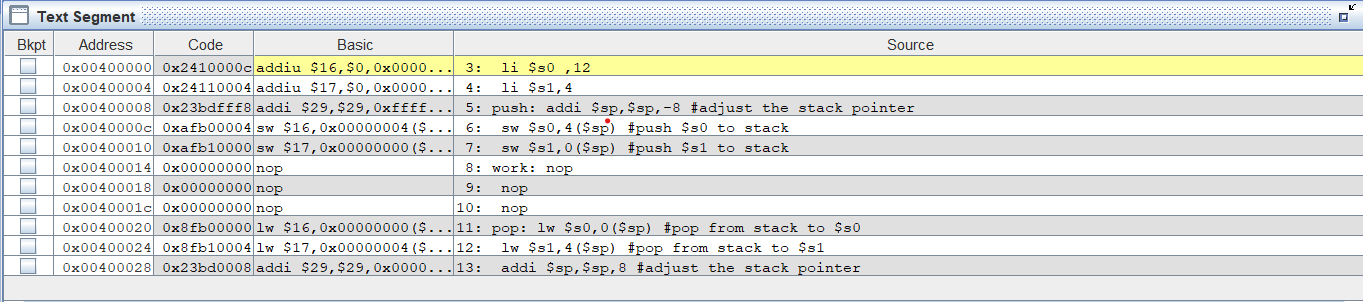
nop

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 thu được**



**Nhận xét :**

-Khi thực hiện lệnh **addi $sp,$sp,-8** trong hàm push thì giá trị của thanh ghi $sp giảm đi 8 từ :

Ảnh có chứa bàn

Mô tả được tạo tự động

Ảnh có chứa bàn

Mô tả được tạo tự động

-Khi thực hiện lệnh **addi $sp,$sp,8** trong hàm pop thì giá trị của thanh ghi $sp tăng lên 8 từ:

Ảnh có chứa bàn

Mô tả được tạo tự động

Ảnh có chứa bàn

Mô tả được tạo tự động

1. **Assignment 4**

**Mã nguồn**

#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

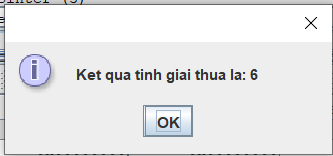
addi $sp,$fp,0 #restore stack pointer

lw $fp,-4($sp) #restore frame pointer

jr $ra #jump to calling

fact\_end:

**Kết quả thu được:**





1. **Assignment 5**

**Mã nguồn**

.text

main:

li $s0, 13

li $s1, 8

li $s2, 9

li $s3, -17

li $s4, 31

li $s5, 33

li $s6, -27

li $s7,5

li $t1,1

li $t2,1

li $t3,1

jal init

nop

li $t4,9

sub $a0, $t4, $t2

sub $a1, $t4, $t3

j end

nop

endmain:

init:

add $v0, $s7, $zero

add $v1, $s7, $zero

push:

addi $sp, $sp, -32

sw $s0,28 ($sp)

sw $s1,24 ($sp)

sw $s2,20 ($sp)

sw $s3,16 ($sp)

sw $s4,12 ($sp)

sw $s5,8 ($sp)

sw $s6,4 ($sp)

sw $s7,0 ($sp)

pop:

addi $sp,$sp,4

lw $a1,0 ($sp)

addi $t1,$t1,1

sub $t0, $a1, $v0

bltz $t0, parel

nop

add $v0, $a1, $zero

add $t2, $t1, $zero

parel:

sub $t0, $a1, $v1

bgtz $t0, pare2

nop

add $v1, $a1, $zero

add $t3, $t1, $zero

pare2:

bne $a1, $s0, pop

nop

done:

jr $ra

end:

Ta đưa vào một mảng A gồm 8 phần tử và gắn lần lượt từ $s0 đến $s7:

**Mảng A**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A[0] | A[1] | A[2] | A[3] | A[4] | A[5] | A[6] | A[7] |
| 13 | 8 | 9 | -17 | 31 | 33 | -27 | 5 |

**Kết quả:**

Ảnh có chứa bàn

Mô tả được tạo tự động

**Largest :$v0,$a0**

**Smallest:$v1,$a1**

Ta thu được :

* Giá trị lớn nhất là 33 tại phần tử thứ 6(A[5])
* Giá trị nhỏ nhất là -27 tại phần tử thứ 7 (A[6])