Báo cáo tuần 7

Phùng Ngọc Vinh – 20194719

Bài 1:

Ảnh có chứa văn bản

Mô tả được tạo tự động

Yêu cầu :

Tính giá trị tuyệt đối của số được truyền vào.

Kết quả:

Đầu vào: $t0 = -45



Đầu ra: $s0 = 45



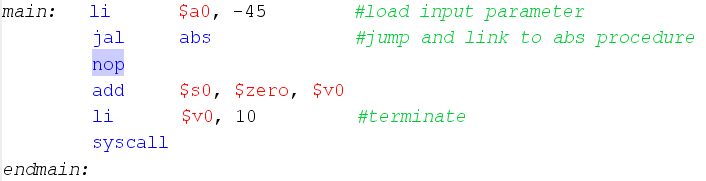
Nhận xét:

Kết quả chạy chương trình chính xác.

Ngoài ra ta thấy thanh ghi kết quả lưu tại thanh ghi $ra:



Đây chính là địa chỉ của dòng lệnh nop trong hàm Main



Mục đích là để quay lại chương trình chính.

Bài 2:Ảnh có chứa văn bản

Mô tả được tạo tự động

Yêu cầu:

Tìm số lớn nhất trong 3 số nguyên được truyền vào.

Kết quả:

Đầu vào: $a0 = 2; $a1 = 6; $a2 = 9



Đầu ra kết quả lưu tại: $s0 = 9



Nhận xét:

Kết quả chạy chương trình chính xác.

Đầu tiên so sánh 2 giá trị của 2 thanh ghi $a0 và $a1. Số lớn hơn lưu tại thanh ghi $s0:

Ảnh có chứa văn bản

Mô tả được tạo tự động

Tiếp theo so sánh $s0 và $a2 để tìm ra số lớn nhất. Kết quả lưu tại thanh ghi $s0:

Ảnh có chứa văn bản

Mô tả được tạo tự động

Sau đó quay trở lại chương trình chính và kết thúc chương trình:



Ảnh có chứa văn bản

Mô tả được tạo tự động

Bài 3: Ảnh có chứa văn bản

Mô tả được tạo tự động

Yêu cầu:

Push lần lượt giá trị của 2 thanh ghi $s0 và $s1 vào Stack.

Pop 2 giá trị đó ra khỏi Stack.

Kết quả:

Đầu vào: $s0 = 26; $s1 = 7



Đầu ra:

Lấy giá trị phần tử trên cùng từ Stack vào thanh ghi $s0:



Lấy giá trị phần tử tiếp theo từ Stack vào thanh ghi $s1:



Nhận xét:

Kết quả chạy chương trình chính xác.

Bài 4:

#Laboratory Exercise 7, Home Assignment 4

.data

Message: .asciiz "Ket qua tinh gia

i 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 $a0,4 #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:

Yêu cầu:

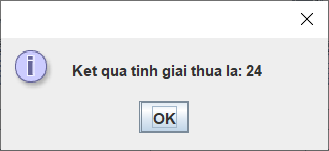
Tính n!

Kết quả:

Đầu vào: $a0 = 4



Đầu ra:



Ảnh có chứa bàn

Mô tả được tạo tự động

Nhận xét:

Kết quả chạy chương trình chính xác.

Với n = 4

0x4, 0x3, 0x2, 0x1 lần lượt là các giá trị $a0 sau mỗi vòng lặp được Stack giữ lại.

Các giá trị khác 0 còn lại theo từng khoảng bộ nhớ là $fp và $ra

Ảnh có chứa bàn

Mô tả được tạo tự động

\*Yêu cầu thêm:

#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:

addi $sp,$sp,-4 #adjust stack pointer (3)

sw $ra,0($sp) #save return address (4)

li $a0,4 #load test input N

jal FACT #call fact procedure

nop lw $ra,0($sp) #restore return address (5)

addi $sp, $sp, 4

jr $ra

wrap\_end:

#----------------------------------------------------------------------

#Procedure FACT: compute N!

#param[in] $a0 integer N

#return $v0 the largest value

#----------------------------------------------------------------------

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

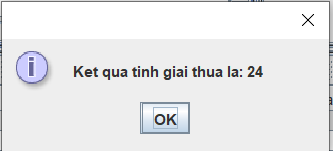
lw $a0,0($sp) #restore a0

addi $sp, $sp, 8

jr $ra #jump to calling

fact\_end:

Kết quả:



Nhận xét:

So với mã nguồn cũ thì mỗi thực thi câu lệnh lw xong t cần chạy thêm các câu lệnh:

addi $sp,$sp,4 và addi $sp,$sp,8

Bài 5:

Code:

.data

MesOfMax: .asciiz "Largest: "

MesOfMin: .asciiz "Smallest: "

Index: .asciiz "$s"

.text

main: li $s0, -100

li $s1, 0

li $s2, 26

li $s3, 7

li $s4, 2001

li $s5, 4

li $s6, -101

li $s7, 100

jal init # call max procedure

nop

li $v0, 4

la $a0, MesOfMax # print max value

add $a1,$t0,$zero

syscall

la $a0, Index

add $a1,$t8,$zero

syscall

la $a0, MesOfMin # print min value

add $a1,$t1,$zero

syscall

la $a0,Index

add $a1,$t9,$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

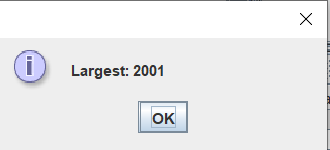
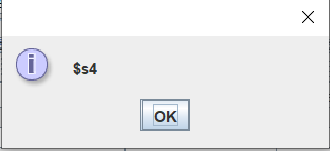
j max\_min # repeat

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

jr $ra # return to calling program

Kết quả:

Max:

Min: