**Báo cáo Thực hành KTMT buổi 7**

**Họ và tên:** Nguyễn Đức Phú

**MSSV:** 20215116

**Assignment 1:**

* Code:

.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] $a0 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:
* Trước khi thực hiện lệnh jal abs



Sau khi thực hiện lệnh



* 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)
* Khi chạy lệnh jr $ra



* Thanh ghi pc trở lại địa chỉ được lưu trong $ra để tiếp tục thực hiện chương trình
* Kết quả sau khi thực hiện xong chương trình:



Giá trị tuyệt đối của $a0 (- 45) đã được lưu vào $s0 (45)

**Assignment 2:**

* Code:

.text

*main:*

li $a0, -27 #load test input

li $a1, 3

li $a2,-3

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

* Kết quả:
* Sự thay đổi của thanh ghi $ra và pc:
  + Khi gọi tới chương trình con max (có địa chỉ 0x00400020)





* + Khi thoát khỏi chương trình con:

Graphical user interface, text, application

Description automatically generated



* 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

* Chương trình con trả về kết quả:



* Là giá trị lớn nhất trong 3 giá trị đầu vào -27, 3, -3

**Assignment 3:**

* Code:

.text

li $s0, 6

li $s1, -9

*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 lệnh addi $sp,$sp,-8 thanh ghi sp trừ đi 8 để chuẩ bị lưu giá trị cho $s0 và $s1





* Sau hai lệnh sw, giá trị $s0 và $s1 được lưu vào stack:

A picture containing chart

Description automatically generated

* Kết quả cuối cùng hai giá trị $s0 và $s1 được đổi chỗ cho nhau



* Thanh ghi sp được trả về giá trị cũ bằng lệnh addi $sp,$sp,8





**Assignment 4:**

* Code:

.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

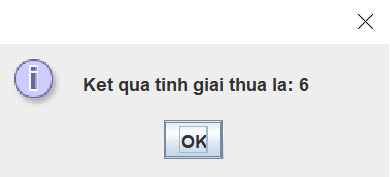
jr $ra #jump to calling

*fact\_end:*

* Kết quả chạy:
* Các giá trị được lưu trong stack:



* Kết quả chạy xong chương trình tính 3!:



* Bảng thể hiện giá trị ngăn xếp:

|  |  |
| --- | --- |
| 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:**

* Code:

.data

largest: .asciiz "Largest: "

smallest: .asciiz "\nSmallest: "

comma: .asciiz ", "

.text

*main:*

li $s0, 3

li $s1, 5

li $s2, -45

li $s3, 6

li $s4, 27

li $s5, -1

li $s6, 666

li $s7, -9

jal saveNumbers

nop

li $v0, 4 # Print message Largest

la $a0, largest

syscall

add $a0, $t0, $zero # Print Max

li $v0, 1

syscall

li $v0, 4 # Print message Comma

la $a0, comma

syscall

add $a0, $t5, $zero

li $v0, 1 # Print the register number of Max

syscall

li $v0, 4 # Print message Smallest

la $a0, smallest

syscall

add $a0, $t1, $zero # Print Min

li $v0, 1

syscall

li $v0, 4 # Print message Comma

la $a0, comma

syscall

add $a0, $t6, $zero

li $v0, 1 # Print the register number of Min

syscall

*endmain:*

li $v0, 10 # Exit

syscall

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

# Return $t0 = Max

# Return $t1 = Min

# Index of Max = $t5

# Index of Min = $t6

#return $v0 the largest value

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

*swapMax:*

add $t0,$t3,$zero

add $t5,$t2,$zero

jr $ra

*swapMin:*

add $t1,$t3,$zero

add $t6,$t2,$zero

jr $ra

*saveNumbers:*

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

*findMaxMin:*

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 # i++

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

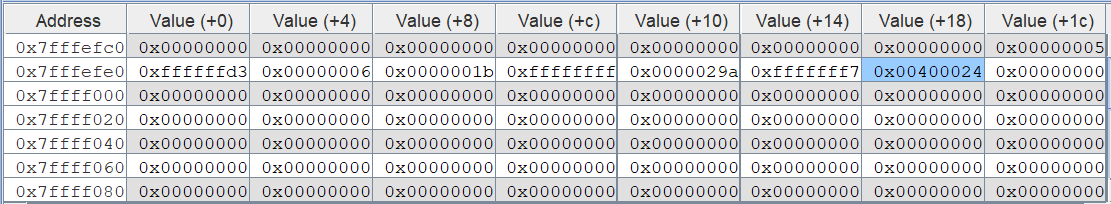
j findMaxMin # Repeat

*done:*

lw $ra, -4($sp)

jr $ra # Return to calling program

* Kết quả chương trình:
* Các giá trị trong ngắn xếp:



* Kết quả cuối:

