**Báo cáo thực hành tuần 7**

Môn học : Thực hành kiến trúc máy tính

Giảng viên : Lê Bá Vui

Sinh viên : Hoàng Đức Anh 20176688

Assignment 1

Code:

#Laboratory Exercise 7 Home Assignment 1  
.text  
main:

li $a0,20 #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

đầu vào $s0=20

Kết quả :



Kết quả chính xác

Assignment 2

Code

#Laboratory Exercise 7, Home Assignment 2  
.text  
main:   
 li $a0,2 #load test input  
 li $a1,10  
 li $a2,5  
 jal max #call max procedure  
 nop  
 add $s0, $zero, $v0  
 li $v0,10 #terminate  
 syscallendmain:  
#----------------------------------------------------------------------  
#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

Đầu vào: $a0=2 $a1=10, $s2=5

Kết quả số lớn nhất:



Thêm lệnh

add $s0, $zero, $v0  
 li $v0,10 #terminate  
 syscall

nếu không sẽ không dừng được chương trình

Assignment 3

Code:

#Laboratory Exercise 7, Home Assignment 3

.text

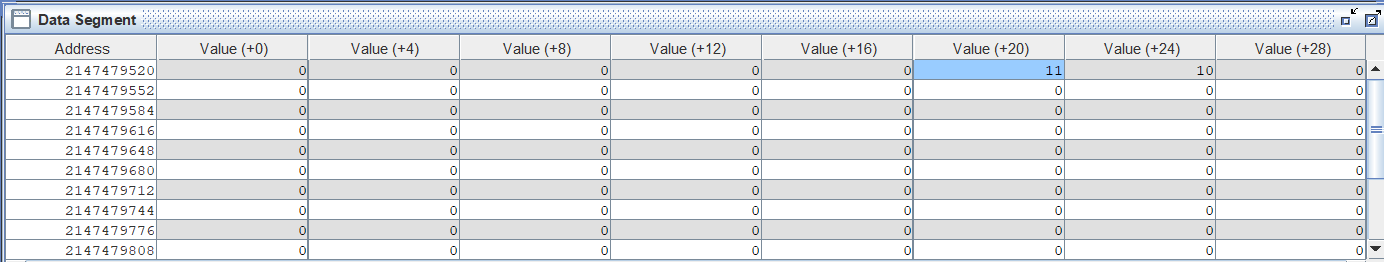
li $s0, 10

li $s1, 11

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

Đầu vào: $s0 = 10, $s1 = 11

Sau khi push:



Giá trị $s0, $s1:



Sau khi pop giá trị $s0, $s1



Giải thích: pop sẽ lấy ra giá trị ở đỉnh ngăn xếp trước và gán vào $s0

Assignment 4

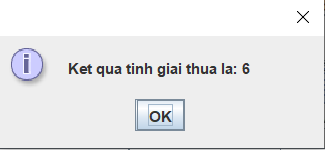
Code:

#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:

Kết quả:



Bảng thể hiện giá trị ngăn xếp

At the begin of WRAP, $sp = 2147479548

|  |  |
| --- | --- |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
| $ra (4) | |
| $fp | |
|  |  |

2147479540 <- new $sp (3) addi $sp,$sp,-8  
2147479544 (1) sw $fp,-4($sp)  
2147479548 🡨 new $fp (2) addi $fp,$sp,0

At the end of WRAP, $sp = 2147479540

|  |  |
| --- | --- |
|  | |
| $a0 |  |
| $ra | |
| $fp | |
| $a0 |  |
| $ra | |
| $fp | |
| $ra |  |
| $fp | |
|  | |

🡨 new $fp

🡨 new $fp

4194308 **🡪**restore $ra **(5)**2147479548**🡪**restore $fp **(7)**2147479540 **🡪**restore $sp **(6)**

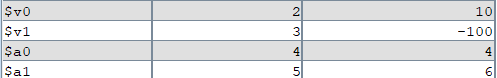
Assignment 5

Code:

.text  
main:  
 li $s0, 2 #load value into register $s0->$s7  
 li $s1, 3  
 li $s2, 4  
 li $s3, 10  
 li $s4, -10  
 li $s5, -100  
 li $s6, 5  
 li $s7, 9  
  
 li $t1,1 # load default value to register $t1  
 li $t2,1  
 li $t3,1  
 jal init # jump to init  
 nop  
 li $t4,9   
 sub $a0,$t4,$t2 # a0= t4-t2  
 sub $a1,$t4,$t3 # a1 = t4-t3  
 j end  
 nop  
endmain:  
init:  
 add $v0,$s7,$zero  
 add $v1,$s7,$zero  
push: # push into stack  
 addi $sp,$sp,-32   
 sw $s0,28($sp) # put turn value of $s0->$ s7 into stack  
 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: # pop out of stack  
 addi $sp,$sp,4  
 lw $a1,0($sp)  
 addi $t1,$t1,1 # loop for  
 sub $t0,$a1,$v0 # t1 = a1-v0  
 bltz $t0,pare1 # branch if less than zero(if t1 <0)  
 nop  
 add $v0,$a1,$zero   
 add $t2,$t1,$zero  
pare1:  
 sub $t0,$a1,$v1 # continue compare  
 bgtz $t0,pare2  
 nop  
 add $v1,$a1,$zero  
 add $t3,$t1,$zero  
pare2:  
 bne $a1,$s0,pop # branch if not equal  
 nop  
done:  
 jr $ra  
 # Largest: $v0,$a0  
 # Smallest: $v1,$a1  
end:

Input: li $s0 = 2   
 $s1 = 3  
 $s2 = 4  
 $s3 = 10  
 $s4 = -10  
 $s5 = -100  
 $s6 = 5  
 $s7 = 9

Kết quả:



Kết quả chính xác

Max : 10 lưu tại thanh ghi số $s3

Min: -100 lưu tại thanh ghi số $s5