Nguyễn Thanh Hiếu
20225716

Week 3

Assignment 1

```
# Laboratory Exercise 7, Assignment 1
.data
	Message: .asciiz "Result: "
.text
main:
	li $a0,-15 #load input parameter
	jal abs		#jump and link to abs procedure
	nop
	add $s0, $zero, $v0		#$a1 = abs($a0)

	li $v0,10	#terminate
	syscall
endmain:

#function abs
#param[in] $a1 the integer need to be gain 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
```

- Nhận xét: Thanh ghi s0 lưu giá trị tuyệt đối của thanh ghi a1

Assignment 2

```
# Laboratory Exercise 7, Assignment 2
.data
        Message: .asciiz "Result: "
.text
main:
        li $a0, 123    #load input
        li $a1, 36
        li $a2, 29
        jal max        #call max procedure
        nop

        add $s0, $zero, $v0

        li $v0, 56
        la $a0, Message
        syscall

        li $v0, 10     #terminate
        syscall
endmain:
#return $v0 the largest value

max:
        add $v0, $a0, $zero
```

```
        sub $t0, $a1, $v0
        bltz $t0, okay
        nop
        add $v0, $a1, $zero
okay:
        sub $t0, $a2, $v0
        bltz $t0, done
        nop
        add $v0, $a2, $zero
done:
        jr $ra
```

- Nhận xét: Thanh ghi s0 lưu giá trị lớn nhất của 3 thanh ghi a0, a1, a2

Assignment 3

```
# Laboratory Exercise 7, Assignment 3
.text
        li $s0,112
        li $s1,36
        jal swap #call max procedure
        nop
        li $v0,10 #terminate
        syscall

#stack: first in last out
swap:
        push:
                addi $sp, $sp, 8      #addjust the stack pointer
                sw $s0, 4($sp)                #s0 -> stack
```

```
        sw $s1, 0($sp)          #s1 -> 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
```

Assignment 4

```
# Laboratory Exercise 7, Assignment 4
.data
    Message: .asciiz "Ket qua tinh gia 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 valua 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  #addjust stack pointer (3)
        sw     $ra, 0($sp)   #save return address (4)

        li     $a0, 6#load test input
        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
WAPRP_END:

#Procedure FACT: compute N!
#Param[in] $a0 interger 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:
```

- Nhận xét:
    - Ảnh minh họa stack với n=3

$a0	Giá trị của FACT(3)
$ra	Địa chỉ của FACT(3)
$fp	
$a0	Giá trị của FACT(2)
$ra	Địa chỉ của FACT(2)
$fp	
$ra	Địa chỉ của FACT(1)
$fp	
$sp	

Assignment 5: Find MAX and MIN procedure

```
# Laboratory Exercise 7, Assignment 5
.data
```

```
        Message1: .asciiz "LARGEST: "
        Message2: .asciiz "SMALLEST: "
        Comma: .asciiz ","
        Endline: .asciiz "\n"
.text
main:
        jal warp
        print:
                add $a1, $v0, $zero # $a1 = result from max(list)
                add $a2, $v1, $zero # $a1 = result from min(list)

                li $v0, 4
                la $a0, Message1
                syscall

                li $v0, 1
                addi $a0, $a1, 0
                syscall

                li $v0, 4
                la $a0, Comma
                syscall

                li $v0, 1
                addi $a0, $t0, 0
                syscall

                li $v0, 4
                la $a0, Endline
```

```
        syscall

        li $v0, 4
        la $a0, Message2
        syscall

        li $v0, 1
        addi $a0, $a2, 0
        syscall

        li $v0, 4
        la $a0, Comma
        syscall

        li $v0, 1
        addi $a0, $t1, 0
        syscall

    quit:
        li $v0, 10 #terminate
        syscall
endmain:

warp:
    addi $fp, $sp, 0
    addi $sp, $sp, -32

    addi $s0, $zero, 12
    sw $s0, 28($sp)
```

```
        addi $s1, $zero, 45
        sw $s1, 24($sp)
        addi $s2, $zero, -52
        sw $s2, 20($sp)
        addi $s3, $zero, -3
        sw $s3, 16($sp)
        addi $s4, $zero, 99
        sw $s4, 12($sp)
        addi $s5, $zero, 18
        sw $s5, 8($sp)
        addi $s6, $zero, -85
        sw $s6, 4($sp)
        addi $s7, $zero, 78
        sw $s7, 0($sp)

        addi $v0, $zero, 0x80000000 # value of max element
        addi $v1, $zero, 0x7fffffff # value of min element
        addi $t0, $zero, 7 # index of max element
        addi $t1, $zero, 7 # index of min element
        addi $t7, $zero, 7 # index

loop:
        lw $t2, 0($sp)

        check_max:
        slt $t3, $v0, $t2 #check: max < current
        beq $t3, $zero, check_min # if max > current then next check
        addi $v0, $t2, 0 #if max < current then update: max = current
        addi $t0, $t7, 0 # update index
```

```
check_min:
slt $t3, $t2, $v1#check: current < min
beq $t3, $zero, continue# if current > min then continue
addi $v1, $t2, 0 #if min > current then update: min = current
addi $t1, $t7, 0 #update idex

continue:
addi $sp, $sp, 4
addi $t7, $t7, -1
bne $sp, $fp, loop
li $fp, 0
jr $ra
```