HW #2

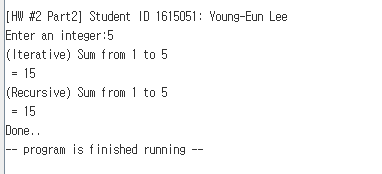
1615051 Young-Eun Lee

**Part2**

**1. MIPS Code**

|  |
| --- |
| # Registers used:  # $v0 - syscall parameter. Returned value  # $a0 - syscall parameter  # $t0 - set if less than 0, syscall parameter  # $t1 – syscall parameter  # $sp - store parameter in stack  # $ra - return address  #Data for the program  .data  my\_info: .asciiz "[HW #2 Part2] Student ID 1615051: Young-Eun Lee\n"  hello\_msg: .asciiz "\nEnter an integer:"  sum\_iter: .asciiz "(Iterative) Sum from 1 to "  sum\_rec: .asciiz "(Recursive) Sum from 1 to "  space: .asciiz " "  end: .asciiz "\nDone.."  equal: .asciiz " = "  next: .asciiz "\n"    # User Program  .text  main:  ## Print hello message  la $a0, my\_info #print 'my\_info'  li $v0, 4 #setup 'print\_string' syscall  syscall #make the syscall  la $a0, hello\_msg # print 'hello\_msg'  li $v0, 4 # setup 'print\_string' syscall  syscall #make the syscall  li $v0, 5 #load syscall read\_int into $v0  syscall #make the syscall  la $a0, sum\_iter #print 'sujm\_iter'  li $v0, 4 #setup 'print\_string' syscall  syscall #make the syscall  li $v0, 5 #load syscall read\_int into $v0  syscall #make the syscall  move $t0, $v0 #move $v0 to $t0  la $a0, equal #print 'equal'  li $v0, 4 #setyp 'print\_string' syscall  syscall #make the syscall  move $a0, $t0 #move $v0 into $a0  jal iter #jump to iter  nop #no operation    la $a0, next #print 'next'  li $v0, 4 #setup 'print\_string' syscall  syscall #make the syscall  la $a0, sum\_rec #print 'sum\_rec'  li $v0, 4 #setup 'print\_string' syscall  syscall #make the syscall  li $v0, 5 #load syscall read\_int into $v0  syscall #make the syscall  la $a0, equal #print 'equal'  li $v0, 4 #setup 'print\_string' syscall  syscall #make the syscall  move $a0, $t1 #move $t1 to $a0  jal rec #jump to rec  nop #no operation  li $v0, 1 #setup 'print\_int' syscall  syscall #make the syscall  la $a0, end #print 'end'  li $v0, 4 #setup 'print\_string' syscall  syscall #make the syscall  b exit #branch to exit  rec:  addi $sp, $sp, -8 # adjust stack for 2 items  sw $ra, 4($sp) # save return address  sw $a0, 0($sp) # save argument  slti $t0, $a0, 1 # $t0=1 if $a0>1  beq $t0, $zero, L1 # n > 1. Call recursive agin  addi $v0, $zero, 0 # if so, result is 0  addi $sp, $sp, 8 # pop 2 items from stack and return  jr $ra #return to call  iter:  beq $a0, $zero, end\_iter #go to end\_iter if $a0 equal 0  add $t1, $t1, $a0 #$t1=$t1+$a0  subi $a0, $a0, 1 #$a0=$a0-1  j iter #jump to iter  L1:  addi $a0, $a0, -1 # else decrement n  jal rec # recursive call  nop ## return point  lw $a0, 0($sp) # restore original n  lw $ra, 4($sp) # and return address  addi $sp, $sp, 8 # pop 2 items from stack  add $v0, $a0, $v0 # $v0=$a0+$v0  jr $ra #return to call    end\_iter:  move $a0, $t1 #move $t1 to $a0  li $v0, 1 #setup 'print\_int' syscall  syscall #make the syscall  jr $ra #return to call  exit:  ## exit the program  li $v0, 10 #setup 'exit' syscall  syscall #make the syscall  ## End of main |

**2. Capture the Console Out**



**3. Code Analysis**

In data area, we are declare the asciiz.(ex my\_info, hello\_msg etc) Next, we can run to program in text area.

First, we are load address data ‘my\_info’, and print to string of ‘my\_info’, syscall. Data ‘hello\_msg’ same algorithm. Second, we are entered the integer (li $v0, 5). And load address data ‘sum\_iter’, and print to string of ‘sum\_iter’, syscall. And again enter the integer(li $v0, 5), syscall. We move $v0 to $t0 register because register is overlap in next line. And load address data ‘equal’ and print to string of ‘equal’, syscall. Next, we move $t0 register to $a0 register. And we jump to iteration lable.

In iteration, if register $a0 equal 0, go to end\_iter label. But if $a0 and 0 is not same, next line. $t1 sum $a0. And this register value go to $t1. $a0 subtract 1. This register value go to $a0. And jump to iter again. This is loop statement. If this iteration is the end, you can go to iteration label ‘end\_iter’. register $t1 move to $a0. And print integer value.

Return to main, we are load address data ‘sum\_rec’, and print to string of ‘sum\_rec’, syscall. And enter the integer(li $v0, 5), syscall. And we move $t1 register to $a0 register. And we jump to recursive label.

In recursive, we adjust stack for 2 items. We save return address in $ra. And we save argument in $a0. $t0 is ‘1’ if $a0 is less than ‘1’. If $t0 is greater than ‘0’, call recursive again. If so, result is 0. Command ‘addi $sp, $sp, 8’ mean pop 2 items from stack and return. and finally return to call. In the recursive label ‘L1’, $a0 add ‘-1’ and this value go to $a0. And next jump to recursive label ‘rec’. And we can restore original entered value. and return address(lw $ra, 4($sp)). Next, we can pop 2 items from stack. And $a0 add to $v0. this register is go to $v0.