CS224
Section No.: 6
Spring 2020
Lab No. 3
Osama Tanveer / 21801147

Part 1

Instruction 1: 0x1109003
Instruction 2: 0x1509FFFA
Instruction 3: 0x08100060
Instruction 4:

lui $1, 0x1001	0x3C011001
ori $8, $1, 0x0088	0x34280088

Instruction 5:

lui $1, 0x1001	0x3C011001
lw $9, 0x0088 ($1)	0x8C290088

Instruction 6:

slt $1, $9, $10	0x012A082A
beq $1, $0, 0xFFF9	0x1020FFF9

Part 2 & 3

```
        .text
.globl __main

__main:

    # first number input
    la $a0, promptForValue1
    li $v0, 4
    syscall
    li $v0, 5
    syscall
    move $s0, $v0

    # second number input
    la $a0, promptForValue2
    li $v0, 4
    syscall
    li $v0, 5
    syscall
    move $s1, $v0

    move $a0, $s0
    move $a1, $s1

    # Calling Fucntion
    jal recursiveMultiplication
```

```
	move $s0, $v0

	# Displaying Product
	la $a0, promptForProductResult
	li $v0, 4
	syscall

	move $a0, $s0
	li $v0, 1
	syscall

	la $a0, endLine
	li $v0, 4
	syscall

	# ADDITION

	# n input
	la $a0, promptForN
	li $v0, 4
	syscall
	li $v0, 5
	syscall
	move $s0, $v0

	move $a0, $s0
	li $a1, 0
	jal recursiveSummation

	move $s0, $v0
	move $a0, $s0
	li $v0, 1
	syscall

	# Exit
	li $v0, 10
	syscall

recursiveMultiplication:
	addi $sp, $sp, -12
	sw $ra, 0($sp)
	sw $a0, 4($sp) # number to add
	sw $a1, 8($sp) # number of times number to add
```

```
        move $v0, $zero

        # base case
        beq $a1, 1, recMulDone

        # find sum of (n-1)
        addi $a1, $a1, -1

        jal recursiveMultiplication
        add $v0, $a0, $v0
        lw $ra, 0($sp)
        lw $a0, 4($sp)
        lw $a1, 8($sp)
        add $sp, $sp, 12

recMulDone:
        jr $ra

recursiveSummation:
        addi $sp, $sp, -8
        sw $ra, 0($sp)
        sw $a1, 4($sp) # number to add
        move $v0, $zero

        # base case
        beq $a1, $a0, recSumDone

        # next term
        addi $a1, $a1, 1

        jal recursiveSummation
        add $v0, $t0, $v0
        lw $ra, 0($sp)
        lw $a1, 4($sp)
        add $sp, $sp, 8
        move $t0, $a1

recSumDone:
        jr $ra

        .data
        promptForValue1: .asciiz "Enter the first number: "
        promptForValue2: .asciiz "Enter the second number: "
```

```
        promptForProductResult: .asciiz "\nThe product is "
        promptForN: .asciiz "Enter the value of n: "
        promptForSumResult: .asciiz "The sum is "
        endLine: .asciiz "\n"
```

Part 4

```
Delete_x:
        addi $sp, $sp, -12
        sw $s0, 0($sp)
        sw $s1, 4($sp)
        sw $s2, 8($sp)

        # a0 -> pointer to the linked list
        # a1 -> x
        li $v0, -1

        # check if head valid
        beq $a0, $zero, headNull
        j headNotNull
headNull:
        j done
headNotNull:
        move $v1, $a0

        move $s0, $a0 # previous
        lw $s1, 0($a0) # current

for:    beq $s1, $zero, doneExceptHead
        lw $s2, 4($s1) # loading value of current
        # checking if value in node equal to given value
        beq $s2, $a1, remove
        move $s0, $s1 # moving previous to current
        lw $s1, 0($s1) # moving current to next
        j for
remove:
        lw $t0, 0($s1)
        sw $t0, 0($s0)
        lw $s1, 0($s1) # moving current to next
        li $v0, 0
        j for
doneExceptHead:
```

```
        lw $t2, 4($v1) # getting value of head
        beq $t2, $a1, removeHead

removeHead:
        lw $t3, 0($v1) # next address
        bne $t3, $zero, newHead
        li $v0, 0
        li $v1, 0
        j done
newHead:
        move $v1, $t3
done:
        lw $s0, 0($sp)
        lw $s1, 4($sp)
        lw $s2, 8($sp)
        addi $sp, $sp, 12

        jr $ra
```