CS 224

Section No: 02
Fall 2019

Lab No: 2

İrem Seven
21704269

Part 1:

a) convertToDec:

```
.data
octalNo: .asciiz "20"

.text
        .globl _start
_start:

#this part calculates the lenght of the string
size:
        lbu $s1,octalNo($s0) #s0: string lenght
        beq $s1,$zero,done
        addi $s0,$s0,1
        j size
        done:

addi $s1,$zero,0 #s1: result
la $a0,octalNo    #a0: adress of octalNo
jal convertToDec
add $a0,$v0,$zero
li $v0,1
syscall

li $v0,10
syscall

convertToDec:
        addi $sp,$sp,-8 #allocating space from stack
        sw $s1,4($sp)   #saving result
        sw $s0,0($sp)   #saving lenght
        addi $t2,$s0,0
        addi $t3,$s0,0
        loop:
                beq $t0,$t2,exit
                lbu $t1,0($a0)
                addi $t1,$t1,-48 #convert ascii to decimal
        octal:
                beq $s0,1,octaldone
                mul $t1,$t1,8
                addi $s0,$s0,-1
                j octal
        octaldone:
        add $s1,$s1,$t1
        addi $a0,$a0,1
        addi $t0,$t0,1
        addi $t3,$t3,-1
        addi $s0,$t3,0
        j loop
        exit:
        addi $v0,$s1,0
        lw $s0,0($sp)
        lw $s1,4($sp)
        addi $sp,$sp,8 #deallocating space
        jr $ra
```

b) interactWithUser:

```
.data
enterNum: .asciiz "Enter an Octal Number: "
erortext: .asciiz "Not an Octal Number\n"
erorlarge: .asciiz "\nNumber cannot include more than 10 digits\n"
octalNo: .ascii
.text
        .globl _start
_start:
#main program
addi $s1,$zero,0        #s1: result
jal interactWithUser    #returns decimal result

li $v0,10
syscall

interactWithUser:
        addi $sp,$sp,-4     #allocating space from stack
        sw $ra,0($sp)
        addi $t1,$t1,0
        la $a0,enterNum
        li $v0,4
        syscall

        li $v0,8            #taking string from user
        la $a0,octalNo
        li $a1,16           #maximum lenght of ascii
        syscall
        addi $s0,$zero,0
        #this part calculates the actual lenght of the string
size:
        lbu $t1,octalNo($s0)     #s0: string lenght
        beq $t1,$zero,done       # if its null
        beq $t1,10,last           #if its end of line
        blt $t1,48,eror           #48: 0 in ascii
        bgt $t1,55,eror          #55: 7 in ascii
        last:
        bgt $s0,10,erorlrg
        addi $s0,$s0,1

        j size
        done:
        addi $s0,$s0,-1 #since ascii sub 1

        j valid
        erorlrg:
                la $a0,erorlarge
                li $v0,4
                syscall
                lw $ra,0($sp)
                addi $sp,$sp,4
            j interactWithUser
```

```
        eror:
                la $a0,erortext
                li $v0,4
                syscall
                lw $ra,0($sp)
                addi $sp,$sp,4
            j interactWithUser
         valid:

        la $a0,octalNo     #a0: adress of octalNo
        jal convertToDec
        add $a0,$v0,$zero  #print the value
        li $v0,1
        syscall
        addi $v0,$a0,0  #return the decimal result
        lw $ra,0($sp)
        addi $sp,$sp,4
        jr $ra


convertToDec:
        addi $sp,$sp,-4  #allocating space from stack
        sw $s1,0($sp)
        addi $t2,$s0,0
        addi $t3,$s0,0
        loop:
                beq $t0,$s0,exit
                lbu $t1,0($a0)
                addi $t1,$t1,-48   #convert ascii to decimal
        octal:
                beq $t2,1,octaldone
                mul $t1,$t1,8   #convert to decimal
                addi $t2,$t2,-1
                j octal
        octaldone:
        add $s1,$s1,$t1
        addi $a0,$a0,1
        addi $t0,$t0,1
        addi $t3,$t3,-1
        addi $t2,$t3,0
        j loop
        exit:
        addi $v0,$s1,0
        lw $s1,0($sp)
        addi $sp,$sp,4     #deallocating space
        jr $ra
```

Part 2: Generating machine instructions

Mips Assembly:	**Machine Code (Hexadecimal):**
beq $t0, $t6, next	0x110E0002
bne $t0, $t6, again	0x150EFFFA
j again	0x08100028

Instructions lw, with label, and la are pseudo instructions.Thus, they first converted into basic assembly.

la $t0, array2

lui	$at, 0x00001001	0x3c011001
ori	$t0, $at, 0x00000064	0x34280064

lw $t1, array2

lui	$at, 0x00001001	0x3c011001
lw	$t1, 0x00000064($at)	0x8C290064

Note:

To not to generate erors, value entered in array1 should not exceed 100 bytes. If it exceeds then address of array2 may be changed implicitly since it is defined after array1. In that case, the machine codes for la and lw may not be provided.