CS224

SECTION 03

LAB02 PRELIMINARY REPORT

AYBÜKE CEREN DURAN/21302686

**1**. **Floating Point Numbers Problem Solving**:

**a-)**  -77.125 in IEEE 754 standard format

Single Precision Format:

Sign: 1

77 in binary form is: 1001101

0.125 in binary form is: 01

77.125 : 1001101.01

When we write 77.125 in scientific form: 1.00110101 x 26

Exponent will become: 6 + 127 = 133

To obtain mantissa, we will take 00110101 part in the scientific form above

Exponent (133) in binary form is : 10000101

Mantissa in binary form is : 00110100100000000000000

Sign Exponent Mantissa

1 10000101 00110100100000000000000

When we convert the result into hex format, we obtain **0xC29A400**

**2. Recursive Summation**

.data

promptSize: .asciiz "Hello! Please enter the size of your number: \n"

enterString: .asciiz "Please enter the string now: \n"

result: .asciiz "Here is the result: \n"

userString: .space 20

.text

entering:

li $v0, 4

la $a0, promptSize

syscall

li $v0, 5

syscall #get the size

li $v0, 4

la $a0, enterString

syscall

li $v0, 8

la $a0, userString

li $a1, 20

syscall #get the string now

move $a3, $a0#user string address is copied into $a3

move $s5, $a0#user string address is copied into $s5

determineSize:

lb $t6, 0($s5)

beq $t6, 0x0A,sizeCalculated

addi $a2, $a2, 1 #increment size

addi $s5, $s5, 1

j determineSize

sizeCalculated:

li $v0, 0

jal recursiveSummation

move $s1, $v0

done:

li $v0, 4

la $a0, result

syscall

li $v0,1

add $a0, $zero, $s1

syscall

li $v0, 10

syscall

#####################

# def digit\_sum(n):

# if n==0 or n==1:

# return n

# else:

# return n+digit\_sum(n-1)

#####################

**recursiveSummation:**

addi $sp, $sp, -12

sw $a2, 8($sp)

sw $a3, 4($sp)

sw $ra, 0($sp)

bgt $a2, 1, sumLoop

lb $t2, 0($a3)

subi $t2, $t2, 48

add $v0, $v0, $t2

addi $sp, $sp, 12

jr $ra

sumLoop:

lb $t2, 0($a3)#get the digit

subi $t2, $t2, 48

add $a3, $a3, 1

subi $a2, $a2, 1

add $v0, $t2, $v0

**jal recursiveSummation**

lw $a2, 8($sp)

lw $a3, 4($sp)

lw $ra, 0($sp)

addi $sp, $sp, 12

jr $ra

**3. deleteAfterX**

**SINCE THE CODE OF LINKED LIST PROGRAM IS TOO LONG I ONLY INCLUDED**

**THE FUNCTION HERE. BUT FOR THE MOSS TEST, I PUT ENTIRE LINKED LIST PROGRAM**

deleteAfter\_x:

lw $t0,4($a0) # get head value

beq $t0,$a1,deleteHead

move $t0,$a0 # t0 = currentPointer

deleteXLoop:

lw $t1,0($t0) # next pointer

beq $t1,0,invalidNumber # if next == NULL

lw $t2,4($t1) # next pointer value

beq $a1,$t2,deleteXLoopEnd

lw $t0,0($t0)

j deleteXLoop

deleteXLoopEnd:

lw $t0, 0($t0) #ben

lw $t1,0($t0)

lw $t1,0($t1)

sw $t1,0($t0)

li $v0, 0

move $v1, $a0

j deleteXLoop#ben

#jr $ra

invalidNumber:

li $v0, -1

move $v1, $a0

jr $ra

deleteHead:

li $v0, 0

move $v1, $a0

lw $v1,0($v1) #head = head->next

jr $ra