**part 1)**

.data

array: .space 80

mess: .asciiz "Enter a number of elements: "

mess2: .asciiz "Enter an element: "

whiteSpace: .asciiz " "

whiteSpace2: .asciiz "\n"

.text

.globl start

start:

li $v0, 9

syscall

la $a0, mess

li $v0, 4

syscall

li $v0, 5

syscall

li $t5, 0 #average t5

beq $v0,$t0, exit

move $t0, $v0

move $s2, $t0 # number of elements s2

sll $t0, $t0, 2

move $t1, $t0

loop:

la $a0, mess2

li $v0, 4

syscall

li $v0, 5

syscall

sw $v0, array($t1)

addi $t1, $t1, -4

bne $t1, $0, loop

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

move $t1, $t0

j display

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

display:

la $a0, whiteSpace

li $v0, 4

syscall

lw $a0, array($t1)

li $v0, 1

syscall

add $t5, $t5, $a0

addi $t1, $t1, -4

bne $t1, $0, display

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

div $t5, $s2

mflo $t5

move $t1, $t0

la $a0, whiteSpace2

li $v0, 4

syscall

j display2

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

display2:

la $a0, whiteSpace

li $v0, 4

syscall

lw $a0, array($t1)

sub $a0, $a0, $t5

li $v0, 1

syscall

addi $t1, $t1, -4

bne $t1, $0, display2

j exit

exit:

li $v0, 10

syscall

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

**part 2)**

.data

charSize: .space 100

notPalindrome: .asciiz "Not a palindrome"

isPalindrome: .asciiz "Is an palindrome"

.text

li $v0, 8 # string read

la $a0, charSize # location

li $a1, 100 # size

syscall

li $t7, 0

la $s0, ($a0)

la $s2, ($a0)

j size

size:

lb $s1, 0($s0)

addi $s0, $s0, 1

addi $t7, $t7, 1 ## char size

bne $s1, 0x000a , size

addi $t7, $t7, -2 ## minus 1 null char

add $s2, $s2, $t7 ## last char

la $s0, ($a0) ## adress again

j isPalin

isPalin:

## first char

lb $s1, 0($s0)

addi $s0, $s0, 1

## sec char

lb $s3 , 0($s2)

addi $s2, $s2, -1

bne $s1, $s3, notPalindrome2

j palindrome

notPalindrome2:

la $a0, notPalindrome

li $v0, 4

syscall

li $v0, 10

syscall

palindrome:

la $a0, isPalindrome

li $v0, 4

syscall

li $v0, 10

syscall

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

**Part 3)**

.data

message: .asciiz "enter a number: "

.text

.globl \_\_start

\_\_start:

la $a0, message

li $v0,4

syscall

li $v0, 5

syscall

move $t0, $v0 # x variable

la $a0, message

li $v0,4

syscall

li $v0, 5

syscall

move $t1, $v0 # y variable

sub $t0, $t0, $t1

sra $t0, $t0, 2

move $a0, $t0

li $v0,1

syscall

li $v0,10

syscall

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

**Part 4)**

Address of z 0x40970040

la $s0, z

lui $at, 0x4097 🡺 0x3C011001 🡺 0011 1100 0000 0001 0001 0000 0000 0001

ori $s0, $at, 0x0040 🡺 0x34300028 🡺 0011 0100 0011 0000 0000 0000 0010 1000

Address of t 0x40970060

la $t1, t

lui $at, 0x4097 🡺 0x3C011001 🡺 0011 1100 0000 0001 0001 0000 0000 0001

ori $t1, $at, 0x0060 🡺 0x3429003C 🡺 0011 0100 0010 1001 0000 0000 0011 1100

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

**Part 5)**

.data

intNumber: .word 0x000000ff

message: .asciiz "The final result is: "

.text

.globl start

start:

addi $t0, $t0, 50

addi $t1, $t1, 4

add $t0, $t0, $t1

sub $t0, $t0, $t1

## division

div $t0, $t1

mflo $t0

mfhi $t1

## mult

mult $t0, $t1

mflo $t0

mfhi $t1

## shifts

sll $t0, $t0, 3

srl $t0, $t0, 3

sra $t0, $t0, 3

## load 32 bit value from mem

lw $t0, intNumber

lui $t1, 0xffff

ori $t1,$t1, 0xfff0

sw $t1, 0($sp)

lw $t1, intNumber

lw $t1, 0($sp)

la $a0,message

li $v0,4

syscall

move $a0, $t0

li $v0,1

syscall