#Write a MIPS program that finds and prints the sum of the main diagonal elements in a 3-by-3 integer matrix A. Assume the matrix A is global and is stored in memory in column-major form. That is, the entries appear in order as: A(0,0), A(1,0), A(2,0), A(0,1), A(1,1), A(2,1), A(0,2), #A(1,2), A(2,2).

#The program

#a. Prompts the user enter 9 integers one by one to fill the matrix A.
#b. Finds the sum of the main diagonal elements in the matrix A
#c. prints the sum value.

.data
A: .space 36 #allocates 36 bytes for the 3x3 integer matrix, each integer needs 4 bytes space

A00: .asciiz "Enter A(0,0): ": 
A10: .asciiz "Enter A(0,1): ": 
A20: .asciiz "Enter A(0,2): ": 
A01: .asciiz "Enter A(1,0): ": 
A11: .asciiz "Enter A(1,1): ": 
A21: .asciiz "Enter A(1,2): ": 
A02: .asciiz "Enter A(2,0): ": 
A12: .asciiz "Enter A(2,1): ": 
A22: .asciiz "Enter A(2,2): ": 

nextLine: .asciiz "\n";
sum: .asciiz "sum of the main diagonal elements in matrix= ": 

.text
.globl main

main:
la $t0, A #loads matrix into base address t0

#Enter Integer A(0,0)
li $v0, 4 #system call code for print_str
la $a0, A00 #address of string to print
syscall #print the string
li $v0, 5 #read user int for A(0,0)
syscall #print
sw $v0, 0($t0) #save it to A

#Enter Integer A(0,1)
li $v0, 4
la $a0, A10
syscall
li $v0, 5
syscall
sw $v0, 4($t0)

#Enter Integer A(0,2)
li $v0, 4
la $a0, A20
syscall
li $v0, 5
syscall
sw $v0, 8($t0)

#Enter Integer A(1,0)
li $v0, 4
la $a0, A01
syscall
li $v0, 5
syscall
sw $v0, 12($t0)

#Enter Integer A(1,1)
li $v0, 4
la $a0, A11
syscall
li $v0, 5
syscall
sw $v0, 16($t0)

#Enter Integer A(1,2)
li $v0, 4
la $a0, A21
syscall
li $v0, 5
syscall
sw $v0, 20($t0)

#Enter Integer A(2,0)
li $v0, 4
la $a0, A02
syscall
li $v0, 5
syscall
sw $v0, 24($t0)

#Enter Integer A(2,1)
li $v0, 4
la $a0, A12
syscall
li $v0, 5
syscall
sw $v0, 28($t0)
# Enter Integer A(2,2)
li $v0, 4
la $a0, A22
syscall
li $v0, 5
syscall
sw $v0, 32($t0)

# SUM
li $v0, 4               # system call code for print_str
la $a0, sum            # address of string to print
syscall                # print string
add $t2,$0,$0          # initialize the sum result to 0
lw $t1, 0($t0)          # loads the element A(0,0) in the main diagonal.
add $t2,$t2,$t1         # sum = sum + A(0,0)
lw $t1, 16($t0)         # loads the element A(1,1) in the main diagonal.
add $t2,$t2,$t1         # sum = sum + A(1,1)
lw $t1, 32($t0)         # loads the element A(0,0) in the main diagonal.
add $t2,$t2,$t1         # sum = sum + A(2,2)
move $a0, $t2           # moves sum result to a0 to print out
li $v0, 1               # system call code for exit
syscall
li $v0, 4               # system call code for print_str
la $a0, nextLine        # address of string to print
syscall                # print string
li $v0, 10              # system call code for exit
syscall
Write a MIPS program that

i. prompts the user to enter a string.

ii. stores the string entered by the user in a global character array.

iii. counts and then prints the number of letters in the set \{a-m, A-M\} and the number of letters in

the set \{n-z, N-Z\} in the string entered by the user.

You can assume that the user will not enter a string more than 15 characters long. You can

assume that the string entered by the user only has lower case or upper case letters and no

other characters.

.data
tenAString: .asciiz "Enter a string: 
numA2M: .asciiz "The number of letters in the set \{a-m, A-M\}= 
numOfN2Z: .asciiz "The number of letters in the set \{n-z, N-Z\}= 
nextLine: .asciiz "\n
string: .space 16

.text
.globl main
main:
add $t0, $0,
#initialize A-M counter to 0
add $t1, $0, $0
#initialize N-Z counter to 0
la $t2, string
#load the base address of string to t2

#print prompt "Enter a string:"
li $v0, 4
la $a0, enterAString
syscall

#get string from console
li $v0, 8
la $a0, string
li $a1, 16
syscall

loop:
lb $t3, 0($t2)
#a = string[i]
beq $t3, $zero, end
#reached Enter, end

li $t4, 77
#M' = 77
ble $t3, $t4, addAM
#string[i] <= 'M', so {a-m, A-M} ++

li $t4, 90
#Z' = 90
ble $t3, $t4, addNZ
#string[i] <= 'Z', so {n-z, N-Z} ++

li $t4, 109
#m' = 109
ble $t3, $t4, addAM
#string[i] <= 'm', so {a-m, A-M} ++
li $t4, 122  #'z' = 122
ble $t3, $t4, addNZ  #string[i] <= 'z', so {n-z,N-Z} ++

j next  #else go to next letter

addAM: addi $t0, 1  #add one to {a-m,A-M}
        j next

addNZ: addi $t1, 1  #add one to {n-z,N-Z}
        j next

#process next character
next:   addi $t2, 1
j loop

end:

#EndDisplay “The number of letters in the set {a-m, A-M}=
li $v0, 4
la $a0, numberOfA2M
syscall

#EndDisplay {a-m,A-M} counter
li $v0, 1
move $a0, $t0
syscall

#EndDisplay new line
li $v0, 4
la $a0, nextLine
syscall

#EndDisplay “The number of letters in the set {n-z, N-Z}=
li $v0, 4
la $a0, numberOfN2Z
syscall

#EndDisplay {n-z,N-Z} counter
li $v0, 1
move $a0, $t1
syscall

#EndDisplay new line
li $v0, 4
la $a0, nextLine
syscall

#Endexit the program
li $v0, 10
syscall