# Author: Casey Bladow

# Date: February 4, 2015

# Program: Asmt2 - 1089 Math Trick

# Source: https://github.com/tandasima/1089-Maths-Trick-MIPS-Assembly

# The amount I dislike this language is incredible.

# Used this source as a guide (tutorial if you will) and followed along...

# I will say that it helped me tremendously in understanding how MIPS works.

.data # variable declarations follow this line

messageInput: .asciiz "Enter a 3 Digit Number: "

messageOriginalDigits: .asciiz "Original Number (a): "

messageReversedOriginalDigits: .asciiz "\nReversed Number (b): "

messageDifferenceOfOriginals: .asciiz "\nDifference|a - b|: "

messageDifferencedReversed: .asciiz "\nReverse of Difference: "

messageSolution: .asciiz "\nFinal Solution: "

input: .word # input is a 3 digit number entered by the user

.text # program instructions begin

main:

li $v0, 4 # loading system call code 4 to $v0 appropriate for printing a string

la $a0, messageInput # loading address for the string to be printed

syscall # calling OS to perform the print operation

la $a0, input # setting $a0 to point to the space allocated for writing a word

la $a1, input # getting the length of the space in $a1 so we can't go over the memory limit

li $v0, 5 # loading system call code 5 for getting a string from the user input into register $v0

syscall # reading register $v0 for op code, sees 5 and asks user to input a string, places string in reference to $a0

add $a0, $0, $v0 # assigning $v0 (user input) to $a0 that will be passed to the

# reverseNum function as an argument.

add $t2, $a0, $0 # storing the user input stored previously in $a0 to $t2 for printing, below.

# beginning to print messageOriginalDigits string

li $v0, 4 # loading system call code 4 to $v0 appropriate for printing a string

la $a0, messageOriginalDigits # loading address for the string to be printed

syscall # calling OS to perform the print operation

# end of print

# beginning to print the original number string

move $a0 , $t2 # copying value from register $t2 to $a0

li $v0, 1 # loading system call code 1 to $v0 appropriate for printing an integer value

syscall # calling OS to print the value in $a0

# end of print

jal reverseNum # calling the reverseNum function

add $s0, $v0, $0 # $s0 is the returned value

# beginning to print messageReversedOriginalDigits string

li $v0, 4 # loading system call code 4 to $v0 appropriate for printing a string

la $a0, messageReversedOriginalDigits # loading address for the string to be printed

syscall # calling OS to perform the print operation

# end of print

# beginning to print the reversed number string

move $a0 , $s0 # copying value from register $s0 to $a0

li $v0, 1 # loading system call code 1 to $v0 appropriate for printing an integer value

syscall # calling OS to print the value in $a0

# end of print

sub $s1, $s0, $t2 # subtracting $0 from $t2 : finding the DIFFERENCE between the ORIGINAL num and the RESERVED number

abs $t3, $s1 # absolute value of the DIFFERENCE ($S1 above) between the original and the reversed

# beginning to print messageDifferenceOfOriginals string

li $v0, 4 # loading system call code 4 to $v0 appropriate for printing a string

la $a0, messageDifferenceOfOriginals # loading address for the string to be printed

syscall # calling OS to perform the print operation

# end of print

# beginning to print the reversed number string

move $a0 , $t3 # copying value from register $s0 to $a0

li $v0, 1 # loading system call code 1 to $v0 appropriate for printing an integer value

syscall # calling OS to print the value in $a0

# end of print

add $a0, $0, $t3 # adding $t3 to $a0. $t3 is the difference between the original number and the

# reverse of the original. it is assigned to $a0 that is the argument to be

# passed to the reverseNum function

jal reverseNum # calling the reverseNum function

add $s0, $v0, $0 # $s0 is the returned value from the function

# beginning to print messageDifferencedReversed string

li $v0, 4 # loading system call code 4 to $v0 appropriate for printing a string

la $a0, messageDifferencedReversed # loading address for the string to be printed

syscall # calling OS to perform the print operation

# end of print

# beginning to print the reversed number string

move $a0 , $s0 # copying value from register $s0 to $a0

li $v0, 1 # loading system call code 1 to $v0 appropriate for printing an integer value

syscall # calling OS to print the value in $a0

# end of print

add $s2, $t3, $s0 # getting the sum of the difference and the reverse of the difference

# beginning to print messageSolution string

li $v0, 4 # loading system call code 4 to $v0 appropriate for printing a string

la $a0, messageSolution # loading address for the string to be printed

syscall # calling OS to perform the print operation

# end of print

# beginning to print the solution string

move $a0 , $s2 # copying value from register $s2 to $a0

li $v0, 1 # loading system call code 1 to $v0 appropriate for printing an integer value

syscall # calling OS to print the value in $a0

# end of print

li $v0, 10 # loading system call code 10 to $v0 appropriate for exiting

syscall # calling OS to exit the program

# VARIABLES $a0 = int num; $s0 = int reverse;

reverseNum:

add $s0 ,$0, $0 # int reverse = 0;

while:

ble $a0, $0 done # if(num <= 0) exit loop ELSE do below while(num > 0) done:

rem $t0, $a0, 10 # $t0 = (num%10)

mul $t1, $s0, 10 # $t1 = (reverse \* 10)

add $s0, $t1, $t0 # reverse = reverse = reverse \* 10 + (num % 10)

div $a0, $a0, 10 # num = num/10

j while

done:

add $v0, $s0, $0 # putting the value of reverse to $v0 for return purposes

jr $ra # returning to caller

