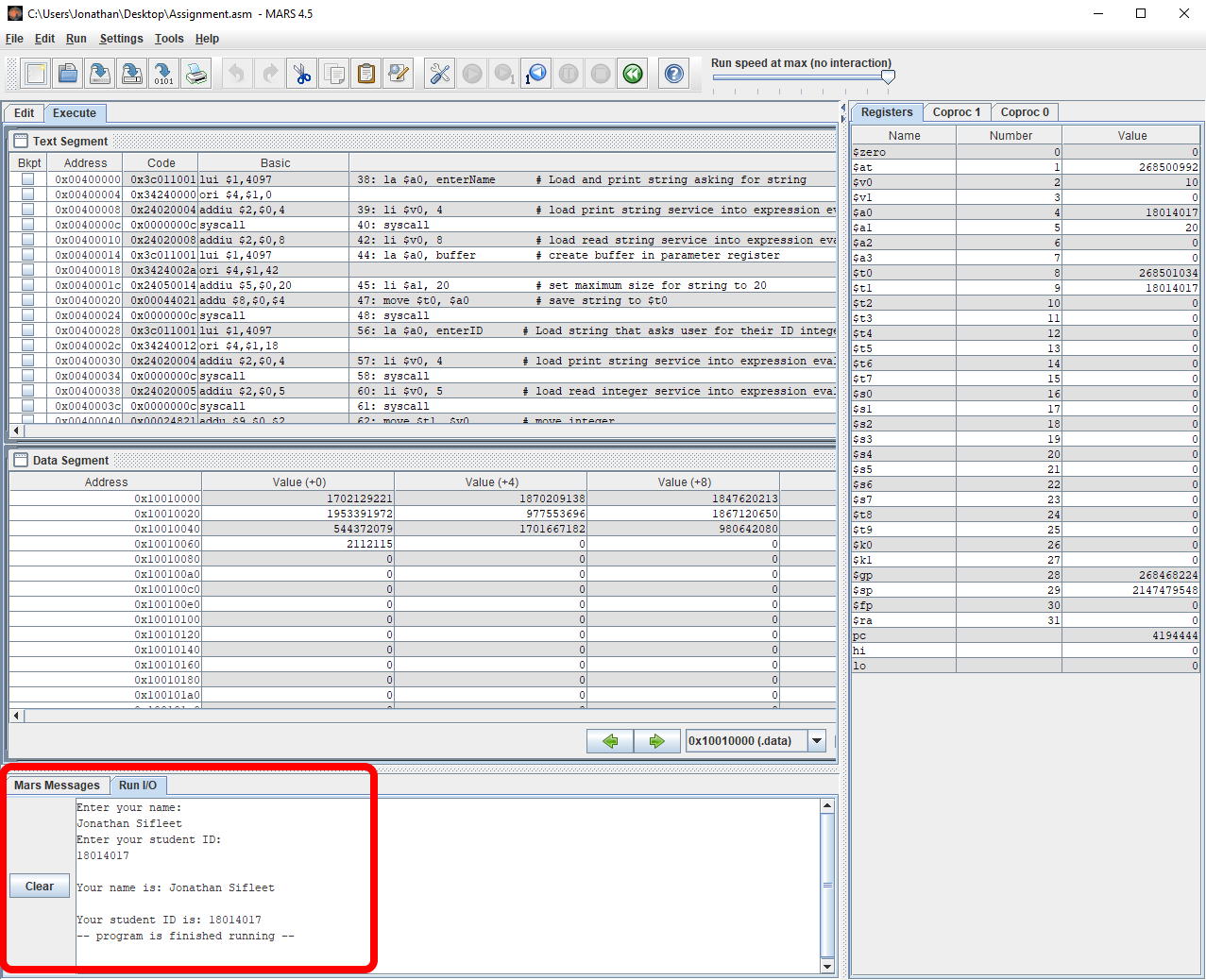
MIPS Assembly Language Assignment

# Task A

Create a fully commented programme that allows a user to input their name (one string) and their student registration number (an integer number) and then output both to screen.

## Explanation:

## Screenshot:



## Code:

#------------------------------

# allows a user to input their name (one string) and their student registration number (an integer

# number) and then output both to screen.

#-------------------------------

.data

enterName: .asciiz "Enter your name (max 20 characters):\n"

enterID: .asciiz "Enter your student ID:\n"

buffer: .space 20 # creates a buffer in memory

outputName: .asciiz "\nYour name is: "

outputID: .asciiz "\nYour student ID is: "

#-------------------------------

#This is the data segment

#-------------------------------

.text

#-------------------------------

#This is the body of the code

#-------------------------------

main:

#-------------------------------

# read in user's name

#-------------------------------

la $a0, enterName # Load and print string asking for string

li $v0, 4 # load print string service into expression evaluation register

syscall

li $v0, 8 # load read string service into expression evaluation register

la $a0, buffer # create buffer in parameter register

la $a1, buffer # set maximum size for string to 20

move $t0, $a0 # save string to $t0

syscall

#-------------------------------

# read in user's studentID

#-------------------------------

la $a0, enterID # Load string that asks user for their ID integer

li $v0, 4 # load print string service into expression evaluation register

syscall

li $v0, 5 # load read integer service into expression evaluation register

syscall

move $t1, $v0 # move integer

#-------------------------------

# print out user's name

#-------------------------------

la $a0, outputName # load and print "Your name is" string

li $v0, 4 # load print string service into expression evaluation register

syscall # print string

la $a0, buffer # reload byte space to primary address

move $a0, $t0 # move contents of t0 to a0

li $v0, 4 # load print string service into expression evaluation register

syscall # print string

#-------------------------------

# print out user's student ID

#-------------------------------

la $a0, outputID # load and print "Your name is" string

li $v0, 4 # load print string service into expression evaluation register

syscall

li $v0, 1 # load print integer service into expression evaluation register

move $a0, $t1 # integer to print

syscall

#-------------------------------

# system call code for exit = 10

#-------------------------------

li $v0, 10 # load terminate service into expression eva;uation register

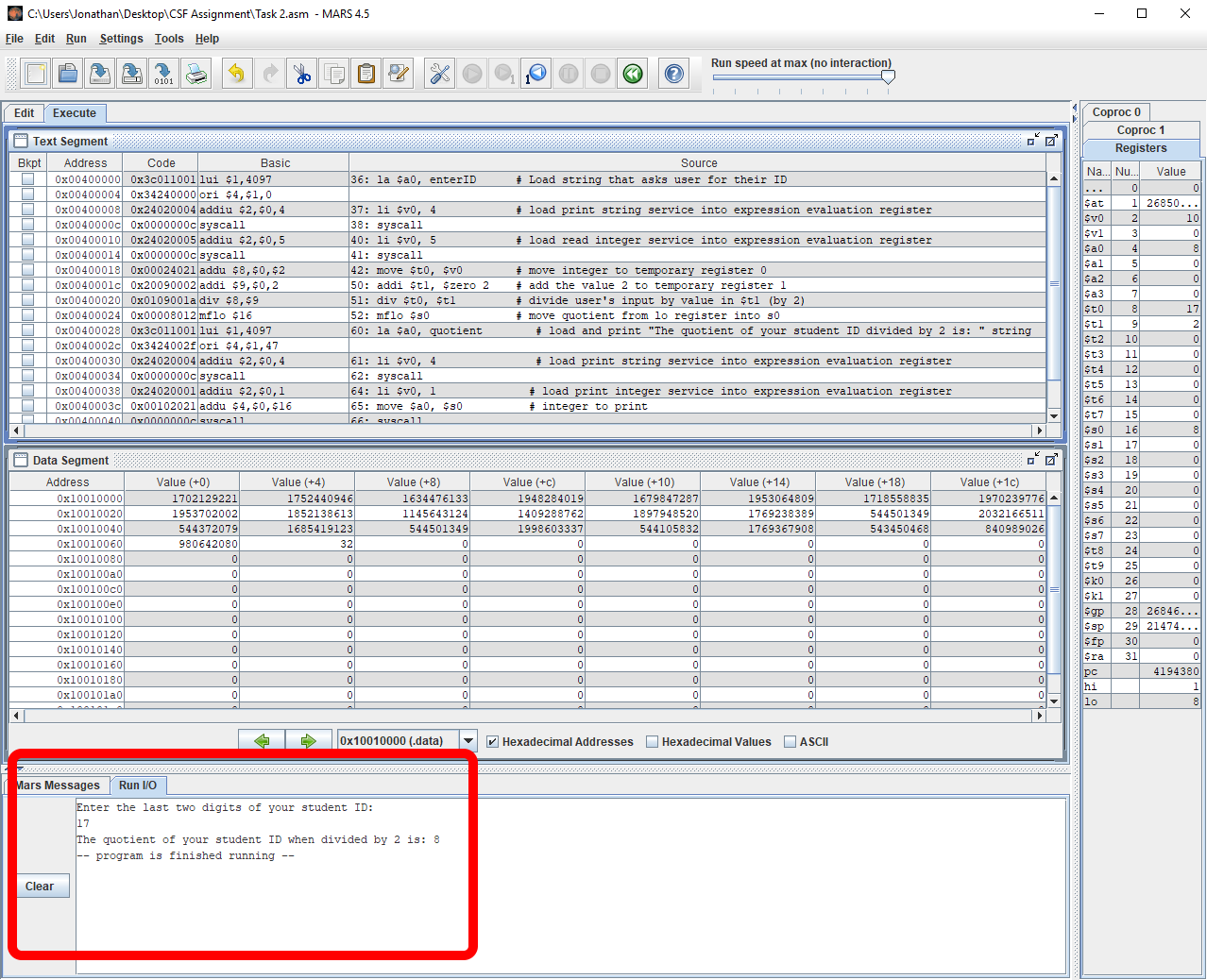
syscall

# Task B

Create a fully commented programme that allows a user to input the last two digits of their student registration number then divides that number by 2 and outputs the nearest quotient INTEGER result to screen.

## Explanation:

## Screenshot:



## Code:

#------------------------------

# Create a fully commented programme that allows a user to input the last

# two digits of their student registration number then divides that number by

# 2 and outputs the nearest quotient INTEGER result to screen.

#-------------------------------

.data

enterID: .asciiz "Enter the last two digits of your student ID:\n"

quotient: .asciiz "The quotient of your student ID when divided by 2 is: "

#-------------------------------

#This is the data segment

#-------------------------------

.text

#-------------------------------

#This is the body of the code

#-------------------------------

main:

#-------------------------------

# read in user's studentID

#-------------------------------

la $a0, enterID # Load string that asks user for their ID

li $v0, 4 # load print string service into expression evaluation register

syscall

li $v0, 5 # load read integer service into expression evaluation register

syscall

move $t0, $v0 # move integer to temporary register 0

#-------------------------------

# divide input by two

#-------------------------------

addi $t1, $zero 2 # add the value 2 to temporary register 1

div $t0, $t1 # divide user's input by value in $t1 (by 2)

mflo $s0 # move quotient from lo register into s0

#-------------------------------

# print result

#-------------------------------

la $a0, quotient # load and print "The quotient of your student ID divided by 2 is: " string

li $v0, 4 # load print string service into expression evaluation register

syscall

li $v0, 1 # load print integer service into expression evaluation register

move $a0, $s0 # move the quotient into the a0 register

syscall

#-------------------------------

# system call code for exit = 10

#-------------------------------

li $v0, 10 # load terminate service into expression evaluation register

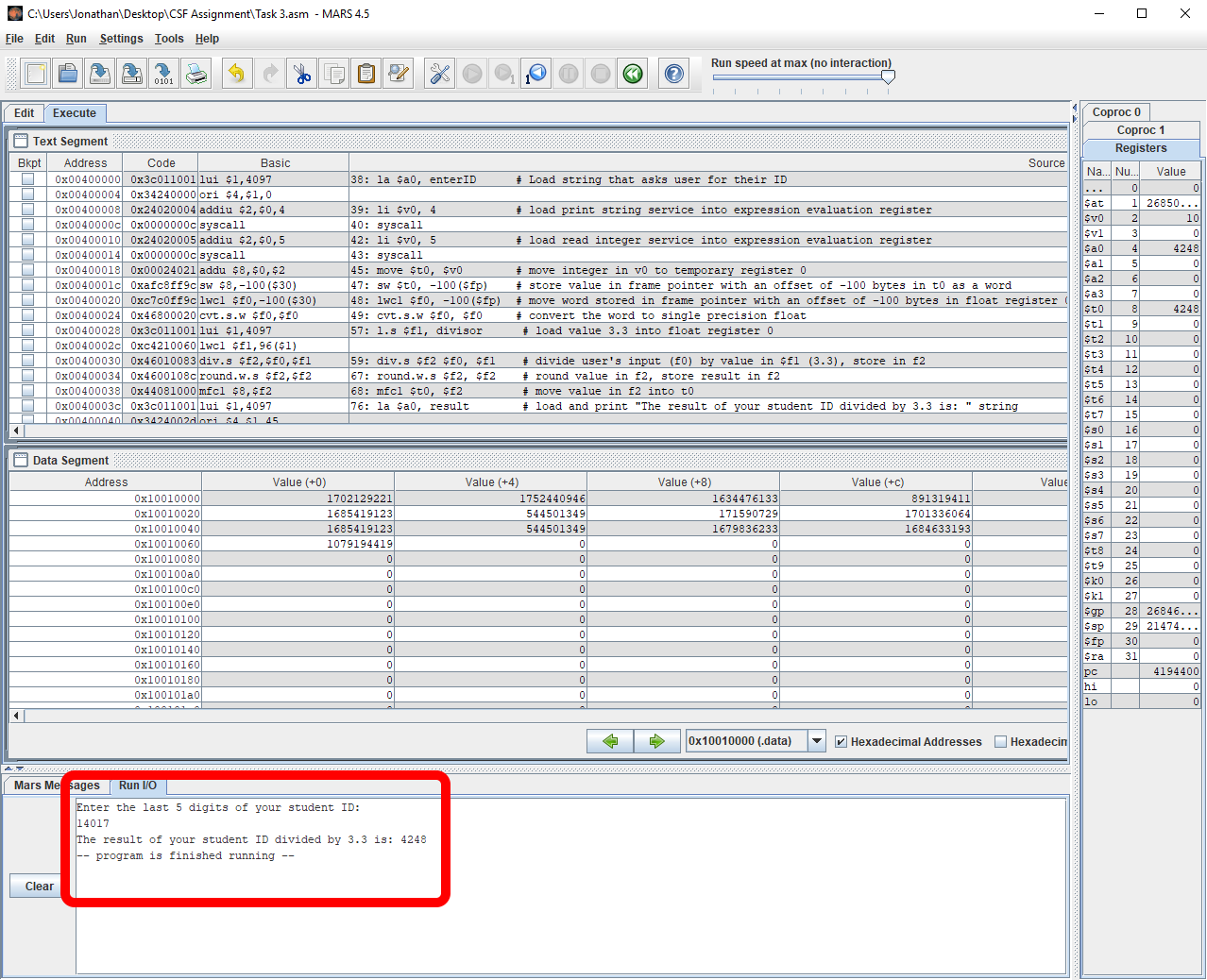
syscall

# Task C

Create a fully commented programme that allows you to input the last five digits of your student registration number (if this results in a four digit number because the leading digit is zero then use the four digit number) then divide that number by 3.3 (three point three) and output the result to screen.

## Explanation:

## Screenshot:



## Code:

#------------------------------

# Create a fully commented programme that allows a user to input the last

# two digits of their student registration number then divides that number by

# 2 and outputs the nearest quotient INTEGER result to screen.

#-------------------------------

.data

enterID: .asciiz "Enter the last 5 digits of your student ID:\n"

result: .asciiz "The result of your student ID divided by 3.3 is: "

divisor: .float 3.3 # stores the value 3.3 as a float

#-------------------------------

#This is the data segment

#-------------------------------

.text

#-------------------------------

#This is the body of the code

#-------------------------------

main:

#-------------------------------

# read in user's studentID

#-------------------------------

la $a0, enterID # Load string that asks user for their ID

li $v0, 4 # load print string service into expression evaluation register

syscall

li $v0, 5 # load read integer service into expression evaluation register

syscall

move $t0, $v0 # move integer in v0 to temporary register 0

sw $t0, -100($fp) # store value in frame pointer with an offset of -100 bytes in t0 as a word

lwc1 $f0, -100($fp) # move word stored in frame pointer with an offset of -100 bytes in float register 0

cvt.s.w $f0, $f0 # convert the word to single precision float

#------------------------------

# divide input by 3.3

#-------------------------------

l.s $f1, divisor # load value 3.3 into float register 0

div.s $f2 $f0, $f1 # divide user's input (f0) by value in $f1 (3.3), store in f2

#-------------------------------

# round value

#-------------------------------

round.w.s $f2, $f2 # round value in f2, store result in f2

mfc1 $t0, $f2 # move value in f2 into t0

#-------------------------------

# print result

#-------------------------------

la $a0, result # load and print "The result of your student ID divided by 3.3 is: " string

li $v0, 4 # load print string service into expression evaluation register

syscall

li $v0, 1 # load print integer service into expression evaluation register

move $a0, $t0 # move the value in t0 to a0, required to print integer

syscall

#-------------------------------

# system call code for exit = 10

#-------------------------------

li $v0, 10 # load terminate service into expression evaluation register

syscall

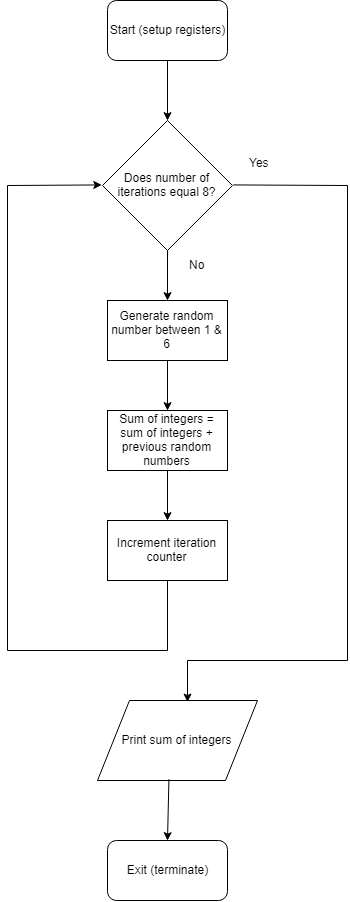
# Task D

A new board game is to be created which requires 8 dice to be thrown simultaneously. All the numbers on the 8 dice are to be added up in order to allow game pieces to move around a board.

## Explanation:

## Screenshots:

## Data Flow Diagram:



## Code

#------------------------------

# Using the “random” Syscall combined with a loop structure, write the MIPs assembly language to

# provided to calculate the sum of 8 random integers

#-------------------------------

.data

#-------------------------------

#This is the data segment

#-------------------------------

.text

#-------------------------------

#This is the body of the code

#-------------------------------

main:

#-------------------------------

# setup registers

#-------------------------------

li $t0 0 # iteration counter

li $t1 0 # sum of all random numbers

#-------------------------------

# generate random numbers

#-------------------------------

li $a1, 6 # load max bound value into a1

li $v0, 42 # load random integer sercice into expression evaluation register

loop:

beq $t0, 8, printInt # if the iteration counter = 8 (number of loops) then print out sum of integers

syscall # generate the random number

add $a0, $a0, 1 # add lowest bound to a0

add $t1, $t1, $a0 # add random int to sum of ints

addi $t0, $t0, 1 # increment loop counter by 1

j loop # jump to top of loop

#-------------------------------

# print sum of random numbers

#-------------------------------

printInt:

move $a0, $t1 # move sum of all random ints (t1) to a0

li $v0, 1 # load print integer service into expression evaluation register

syscall

#-------------------------------

# system call code for exit = 10

#-------------------------------

li $v0, 10 # load terminate service into expression evaluation register

syscall