**FORMAN CHRISTIAN COLLEGE (A CHARTERED UNIVERSITY)**

****

**Computer Organization and Assembly Language – COMP 300 B**

**Spring 21**

**Lab - 04**

**Muhammad Sameed Gilani - 231488347**

**You should attach the lab / assignment handout as second page of this report.**

**From third page onwards following headings should be included:**

* **Introduction**
  + **Should carry information of all major library functions.**
* **Your logic / algorithm in simple English. Bullet points are appreciated.**
* **Your code**
* **Screen shots of at least three outputs of your code with appropriate inputs.**
* **References**

**INTRODUCTION**

* li – Load immediate. 🡪 It is used to set the register to the immediate value we enter.

Ex:

li $v0,1  
This sets the register $v0, to 1

* la – Load address 🡪 It is used to set the register to the contents of another register or to an immediate value we enter.

Ex:

la $a0,$t0  
This loads the contents of $t0 onto $a0

* lw - Load Word 🡪 Set a register to contents of effective memory word address,

Ex:

lw $a0,input

This loads the address of the .word input, we created in the data segment.

* .asciiz 🡪 Store the string in the Data segment and add null terminator. Used in the program to store strings.

Ex:

x: .asciiz " Enter a value for x: "

In the data segment, this string is stored in x.

* .word 🡪 Store the listed value(s) as 32 bit words on word boundary

Ex:

c: .word 3

In the data segment, 3 is stored in c.

* .space 🡪 Reserve the next specified number of bytes in Data segment. It is used in the program to assign specific space for the input user will enter.

Ex:

input1: .space 8

In the data segment, assigns space to the input.

* move 🡪 Move the contents of one register to another

Ex:

move $t0,$t1

Contents of $t1 are moved to $t0

* mul 🡪 Used to multiply the values in 2 registers and store it in a register

Ex:

mul $t0, $t1, $t2

$t1 and $t2 are multiplied and answer is stored in $t0

* add 🡪 Used to add the values in 2 registers and store it in a register

Ex:

add $t0, $t1, $t2

$t1 and $t2 are added and answer is stored in $to

|  |  |  |
| --- | --- | --- |
| print integer | 1 | $a0 = integer to print |

* Service numbers used are,
  + 1 🡪

|  |  |  |
| --- | --- | --- |
| print string |  | $a0 = address of null-terminated string to print |

* + 4 🡪

|  |  |  |  |
| --- | --- | --- | --- |
| read integer |  |  | $v0 contains integer read |

* + 5 🡪

|  |  |  |
| --- | --- | --- |
| read string |  | $a0 = address of input buffer $a1 = maximum number of characters to read |

* + 8 🡪
  + 10 🡪 exit (terminate execution)

**LOGIC**

*Task1:*

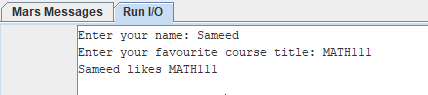
* Created input statements for name and course. And stored the strings, “ like “ and “\n” respectively
* Created input and inputsize variables, to store the name and course input.
* First prompted user to enter their name and stored it in the variable input1
* Then prompted to enter their favourite course and stored it in input2.
* Printed the nextline variable where necessary to move the proceeding string to the next line.
* Once I had both the name and course title. I simply printed their name, then the “ like “, string mentioned earlier. and then their course title.

*Task2:*

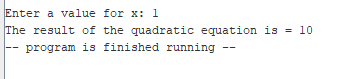
* Created variables for a, b and c as 5, 2 and 3, respectively.
* Created an input statement for x and a final answer statement.
* First prompted the user to enter a value for x.
* Then moved the value of x, a, b and c to separate registers.
* Performed the arithmetic operations. Calulating x2, then ax2.
* Calculated, bx. Then, ax2 + bx
* Finally calculated, ax2 + bx + c. Storing the final answer in the register I’ll refer to as Y.
* Printed the final answer statement and then the final answer, Y. Which was just calculated.
* Finally ending the program properly.

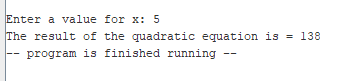
**SAMPLE OUTPUTS**

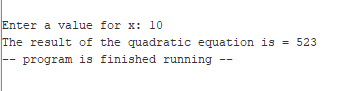
TASK 1:



TASK 2:







**CODE**

.data

# Input statements for users name & course

name: .asciiz "Enter your name: "

course: .asciiz "Enter your favourite course title: "

like: .asciiz " likes "

nextline: .asciiz "\n"

input1: .space 8 # User name input

input2: .space 9 # User fav course input

inputsize1: .word 7 # Name input size

inputsize2: .word 8 # Course title input size

.text

Task1:

# User is asked to input name

li $v0,4

la $a0,name

syscall

# The input is stored in the variable input1

li $v0,8

la $a0,input1

lw $a1,inputsize1

syscall

# Goes to next line

li $v0,4

la $a0,nextline

syscall

# User is asked to input name

li $v0,4

la $a0,course

syscall

# The input is stored in the variable input2

li $v0,8

la $a0,input2

lw $a1,inputsize2

syscall

# Goes to next line

li $v0,4

la $a0,nextline

syscall

# Prints the users name

li $v0,4

la $a0,input1

syscall

# Prints "like" inbetween the name and course

li $v0,4

la $a0,like

syscall

# Prints the course title

li $v0,4

la $a0,input2

syscall

.data

# Created vairbles for each integer

a: .word 5

b: .word 2

c: .word 3

x: .asciiz "\n\nEnter a value for x: " # User input for X

ans: .asciiz "The result of the quadratic equation is = " # Answer statement

.text

Task2:

# Asks user to input an integer, x

li $v0,4

la $a0,x

syscall

# x is stored in $v0 and then moved to $t3

li $v0,5

syscall

move $t3,$v0

# The initially created variables are moved to reigtsers

lw $t0,a

lw $t1,b

lw $t2,c

# Arithmetic operations are performed

mul $t4,$t3,$t3 # x^2 is computed

mul $t5,$t4,$t0 # x^2 \* a is computed and stored in $t5

mul $t6,$t1,$t3 # x \* b is computed and stored in $t6

add $t7,$t6,$t5 # (x^2 \* a) + (x \* b) is computed and stored in $t7

add $t8,$t7,$t2 # (x^2 \* a) + (x \* b) + c , is computed and stored in $t8. i.e y

# Answer statement is printed

li $v0,4

la $a0,ans

syscall

# Final answer is printed

li $v0,1

move $a0,$t8

syscall

# Program ends properly

li $v0,10

syscall