Computer Organization & Architecture Lab

Lab Report # 04

Submitted By: AWAIS SADDIQUI

Registration No: 21PWCSE1993

Section: “A”

“On my honor, as student at University of Engineering and Technology, I have neither given nor received unauthorized. assistance on this academic work”

Student Signature: ______________

Submitted to:

Dr. Bilal Habib

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar.

ASSESSMENT RUBRICS COA LABS

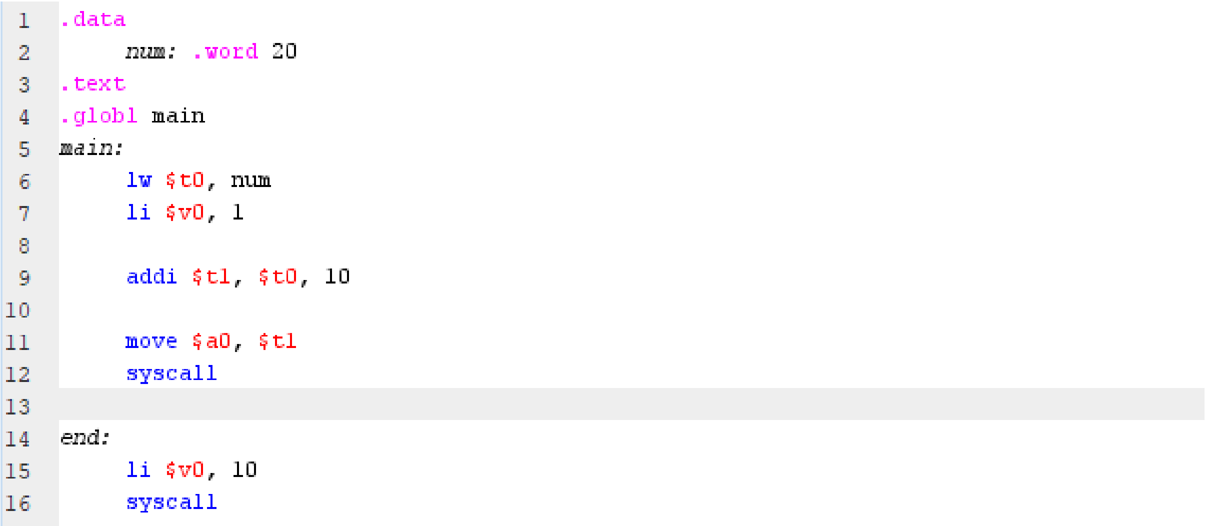
LAB REPORT ASSESSMENT				
Criteria	**Excellent**	**Average**	**Nill**	**Marks Obtained**
1. Objectives of Lab	All objectives of lab are properly covered [Marks 10]	Objectives of lab are partially covered [Marks 5]	Objectives of lab are not shown [Marks 0]	
2. MIPS instructions with Comments and proper indentations.	All the instructions are well written with comments explaining the code and properly indented	Some instructions are missing are poorly commented code [Marks 10]	The instructions are not properly written [Marks 0]	
3. Simulation run without error and warnings	The code is running in the simulator without any error and warnings [Marks 10]	The code is running but with some warnings or errors. [Marks 5]	The code is written but not running due to errors [Marks 0]	
4. Procedure	All the instructions are written with proper procedure	Some steps are missing [Marks 10]	steps are totally missing [Marks 0]	
5. OUTPUT	Proper output of the code written in assembly [Marks 20]	Some of the outputs are missing [Marks 10]	No or wrong output [Marks 0]	
6. Conclusion	Conclusion about the lab is shown and written [Marks 20]	Conclusion about the lab is partially shown	Conclusion about the lab is not shown[Marks0]	
7. Cheating			Any kind of cheating will lead to 0 Marks	
Total Marks Obtained:___________ Instructor Signature: ________________________				

DATA TRANSFER IN MIPS

Task 01:

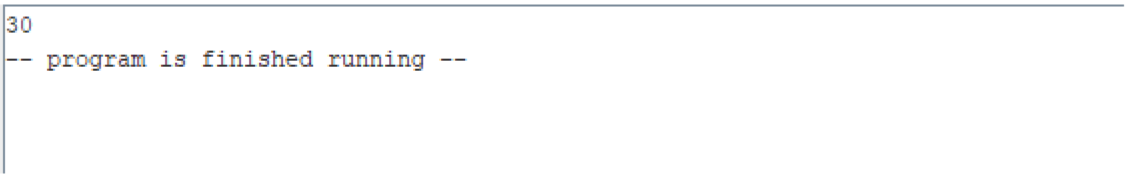
Load a value from memory and add 10 to it. Store the result back in memory and show the result on console.

Code:



```
.data
     num: .word 20
.text
.globl main
main:
     lw $t0, num
     li $v0, 1

     addi $t1, $t0, 10

     move $a0, $t1
     syscall

end:
     li $v0, 10
     syscall
```

Output:

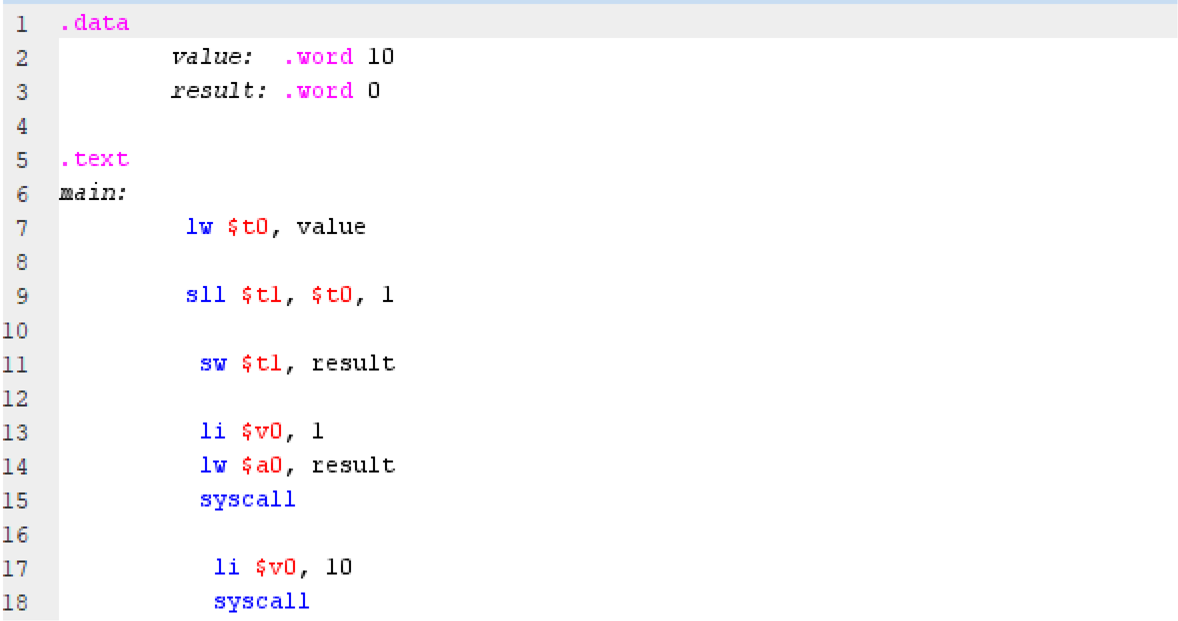


```
30
-- program is finished running --
```

Task 02:

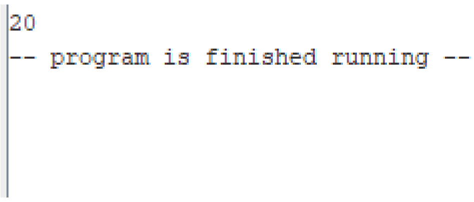
Load a value from memory and double it. Store the result back in memory also show on the console.

Code:



```
.data
        value:  .word 10
        result: .word 0

.text
main:
         lw $t0, value

         sll $t1, $t0, 1

          sw $t1, result

          li $v0, 1
          lw $a0, result
          syscall

           li $v0, 10
           syscall
```

Output:

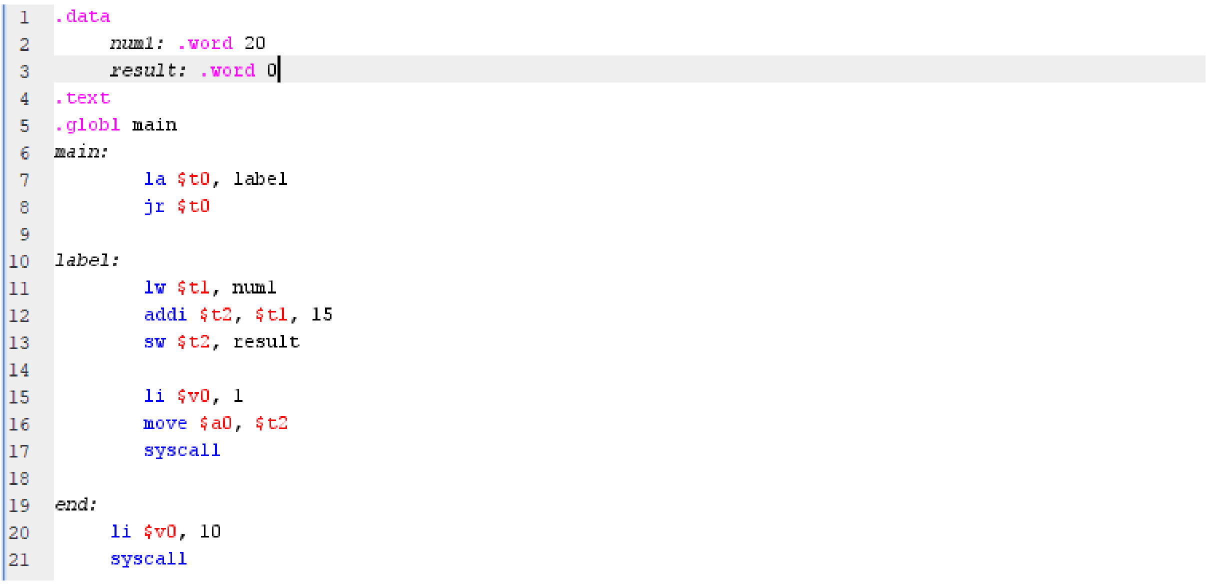


```
20
-- program is finished running --
```

Task 03:

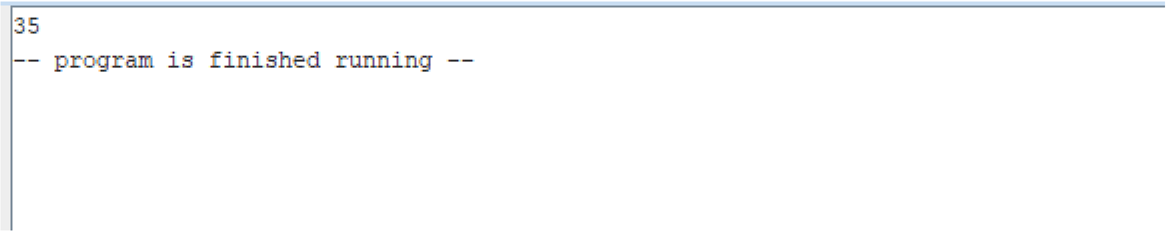
Load an address of a label into a register and jump to that address and perform addition in that address.

Code:



```
.data
     num1: .word 20
     result: .word 0
.text
.globl main
main:
        la $t0, label
        jr $t0

label:
        lw $t1, num1
        addi $t2, $t1, 15
        sw $t2, result

        li $v0, 1
        move $a0, $t2
        syscall

end:
     li $v0, 10
     syscall
```

Output:

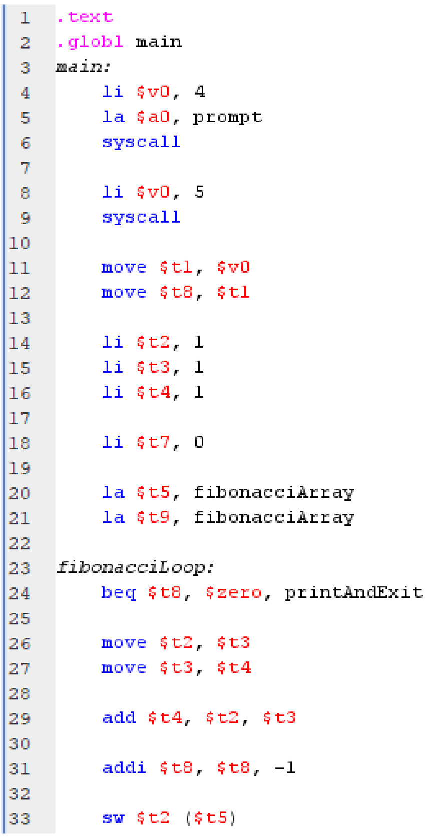


```
35
-- program is finished running --
```

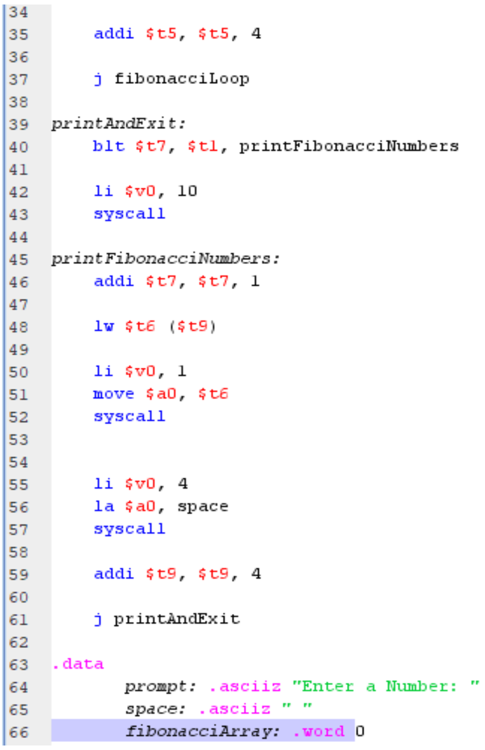
Task 04:

Write assembly program to find the Fibonacci series.

Code:

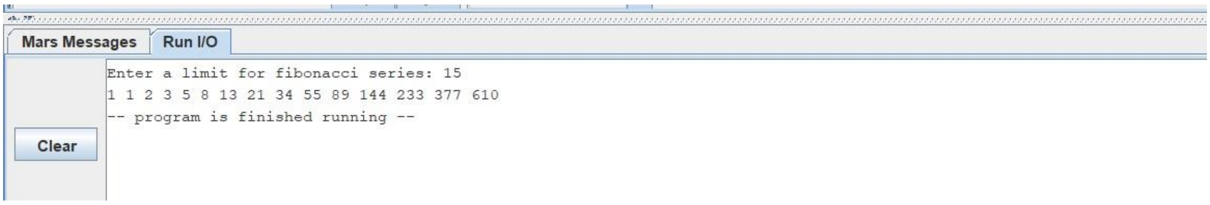


```
.text
.globl main
main:
    li $v0, 4
    la $a0, prompt
    syscall

    li $v0, 5
    syscall

    move $t1, $v0
    move $t8, $t1

    li $t2, 1
    li $t3, 1
    li $t4, 1

    li $t7, 0

    la $t5, fibonacciArray
    la $t9, fibonacciArray

fibonacciLoop:
    beq $t8, $zero, printAndExit

    move $t2, $t3
    move $t3, $t4

    add $t4, $t2, $t3

    addi $t8, $t8, -1

    sw $t2 ($t5)
```



```

    addi $t5, $t5, 4

    j fibonacciLoop

printAndExit:
    blt $t7, $t1, printFibonacciNumbers

    li $v0, 10
    syscall

printFibonacciNumbers:
    addi $t7, $t7, 1

    lw $t6 ($t9)

    li $v0, 1
    move $a0, $t6
    syscall


    li $v0, 4
    la $a0, space
    syscall

    addi $t9, $t9, 4

    j printAndExit

.data
      prompt: .asciiz "Enter a Number: "
      space: .asciiz " "
      fibonacciArray: .word 0
```

Output:



```
Enter a limit for fibonacci series: 15
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
-- program is finished running --
```