

# LAB #04 Report

Name:黄家睿

ID number:202283890036

Major:IOT

## Task1:

1. Write a program that asks the user to enter an integer and then displays the number of 1's in the binary representation of that integer. For example, if the user enters 9, then the program should display 2.

**Solution:**

**Code:**

```
1 .data
2
3 str1: .ascii "Please enter a integer: "
4 str2: .ascii "Number of 1's in binary representation: "
5
6 .globl main
7 .text
8 main:
9     #input the integer
10    li $v0, 4
11    la $a0, str1
12    syscall
13
14    #read the input integer
15    li $v0, 5
16    syscall
17    move $t0, $v0    #save inout integer into $t0
18
19    li $t1, 0        #intialize a counter for the number of 1 is zero
20
```

```
21 check_loop:
22
23     andi $t2, $t0, 1
24     beqz $t2, shift_loop
25     addi $t1, $t1, 1
26
27 shift_loop:
28     srl $t0, $t0, 1
29     bnez $t0, check_loop
30
31     li $v0, 4
32     la $a0, str2
33     syscall
34
35     li $v0, 1
36     move $a0, $t1
37     syscall
38
39     li $v0, 10
40     syscall
```

Input 9 and result is 2:

```
Please enter a integer: 9
Number of 1's in binary representation: 2
— program is finished running —
```

## Tast2:

- Write a program that asks the user to enter two integers: n1 and n2 and prints the sum of all numbers from n1 to n2. For example, if the user enters n1=3 and n2=7, then the program should display the sum as 25.

Solution:

Code:

```
Lab04.asm Lab04_tast2.asm mips3.asm
1 .data
2     input_prompt_n1: .asciiz "Enter the first integer (n1): "
3     input_prompt_n2: .asciiz "Enter the second integer (n2): "
4     output_result:   .asciiz "Sum of numbers from n1 to n2: "
5
6 .text
7     # Prompt the user for n1
8     li $v0, 4
9     la $a0, input_prompt_n1
10    syscall
11
12    # Read n1 from the user
13    li $v0, 5
14    syscall
15    move $t0, $v0 # Store n1 in $t0
16
17    # Prompt the user for n2
18    li $v0, 4
19    la $a0, input_prompt_n2
20    syscall
21
22    # Read n2 from the user
23    li $v0, 5
24    syscall
25    move $t1, $v0 # Store n2 in $t1
26
27    # Initialize the sum to 0
28    li $t2, 0
29
```

```
30    # Loop to calculate the sum
31    sum_loop:
32        add $t2, $t2, $t0 # Add n1 to the sum
33        addi $t0, $t0, 1 # Increment n1
34
35    # Check if n1 is greater than n2
36    bgt $t0, $t1, sum_done
37
38    # Repeat the loop
39    j sum_loop
40
41    sum_done:
42    # Display the result
43    li $v0, 4
44    la $a0, output_result
45    syscall
46
47    # Display the sum (in $t2)
48    li $v0, 1
49    move $a0, $t2
50    syscall
51
52    # Exit the program
53    li $v0, 10
54    syscall
```

Test & screenshot:

Input 3 and 7, the output is 25:

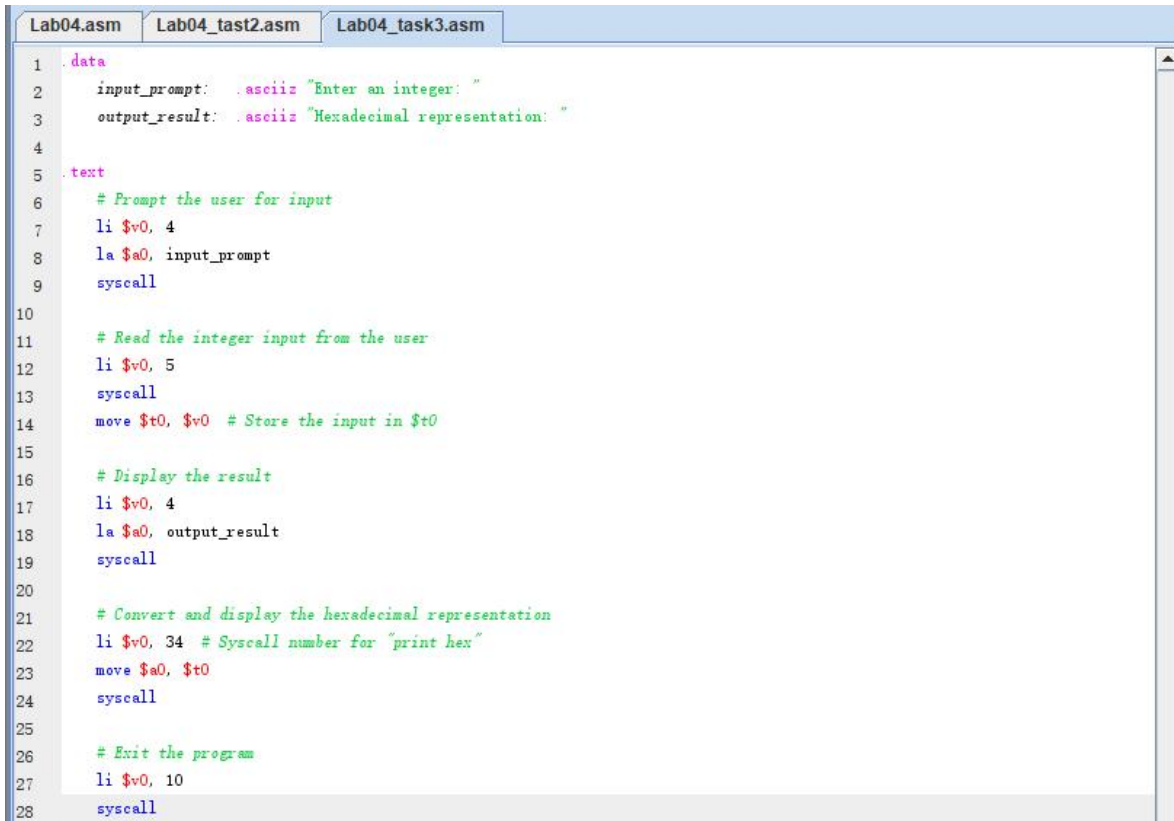


## Tast3:

3. Write a program that asks the user to enter an integer and then display the hexadecimal representation of that integer.

Solution:

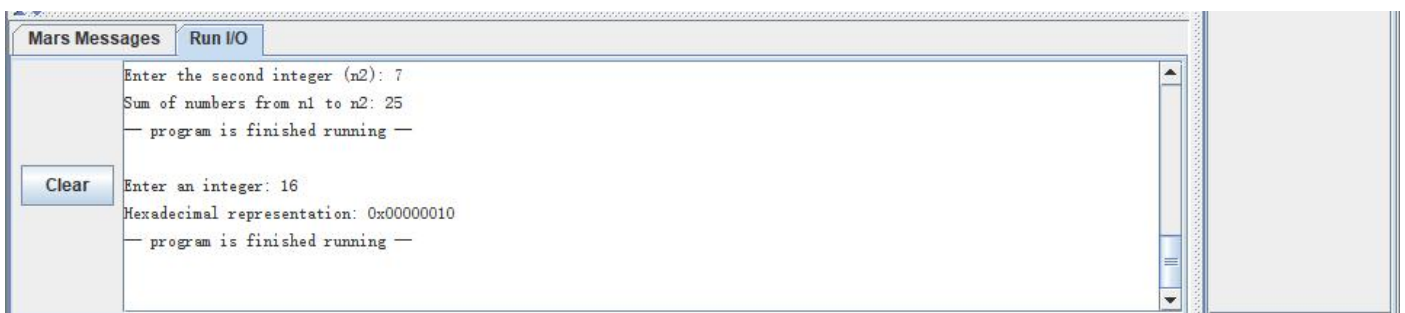
Code:



```
1 .data
2     input_prompt: .asciiz "Enter an integer: "
3     output_result: .asciiz "Hexadecimal representation: "
4
5 .text
6     # Prompt the user for input
7     li $v0, 4
8     la $a0, input_prompt
9     syscall
10
11     # Read the integer input from the user
12     li $v0, 5
13     syscall
14     move $t0, $v0 # Store the input in $t0
15
16     # Display the result
17     li $v0, 4
18     la $a0, output_result
19     syscall
20
21     # Convert and display the hexadecimal representation
22     li $v0, 34 # Syscall number for "print hex"
23     move $a0, $t0
24     syscall
25
26     # Exit the program
27     li $v0, 10
28     syscall
```

Text & screenshot:

Input 16 and the result should is 0x00000010



Mars Messages    Run I/O

Enter the second integer (n2): 7  
Sum of numbers from n1 to n2: 25  
— program is finished running —

Clear

Enter an integer: 16  
Hexadecimal representation: 0x00000010  
— program is finished running —

## Tast4:

4. The Fibonacci sequence are the numbers in the following integer sequence: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...

Write a program that asks the user to enter a positive integer number  $n$  then prints the  $n$ -th Fibonacci number. The following algorithm can be useful:

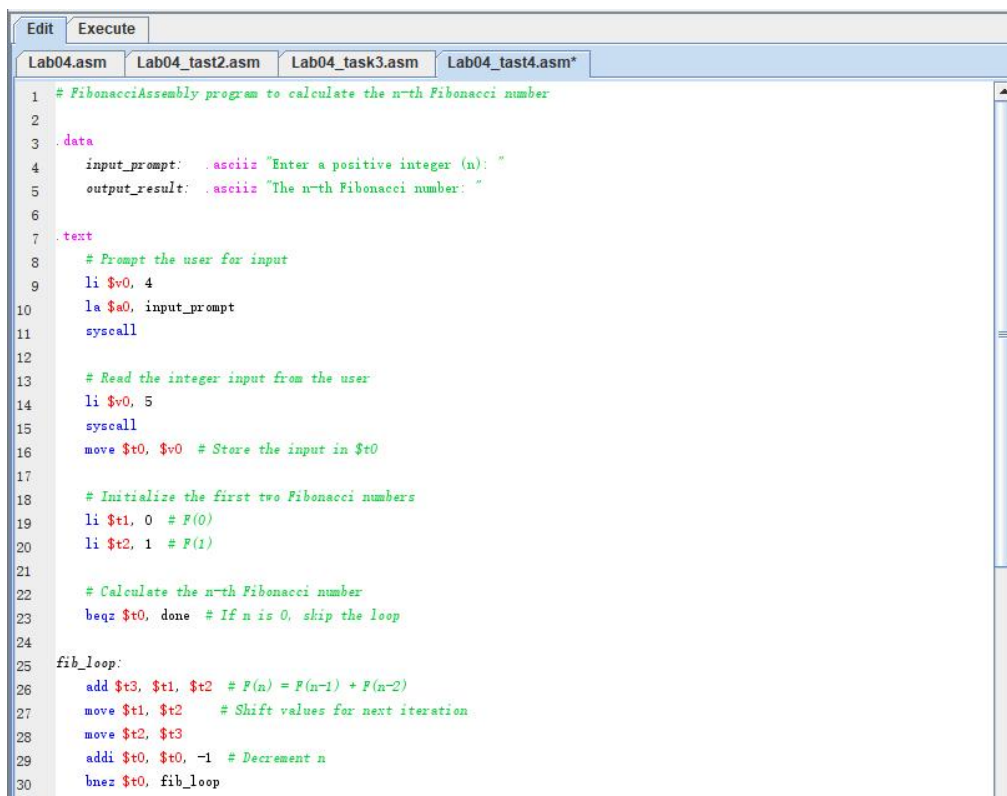
**Input:**  $n$  positive integer

**Output:**  $n$ -th Fibonacci number

Pseudocode
<pre>Fib0 = 0, Fib1 = 1 for (i=2; i&lt;=n; i++) do     temp = fib0     fib0 = fib1     fib1 = temp + fib1 if (n &gt; 0)     fib = fib1 else fib = 0</pre>

**Solution:**

**Code:**



```
1  # FibonacciAssembly program to calculate the n-th Fibonacci number
2
3  .data
4      input_prompt: .asciiz "Enter a positive integer (n): "
5      output_result: .asciiz "The n-th Fibonacci number: "
6
7  .text
8      # Prompt the user for input
9      li $v0, 4
10     la $a0, input_prompt
11     syscall
12
13     # Read the integer input from the user
14     li $v0, 5
15     syscall
16     move $t0, $v0 # Store the input in $t0
17
18     # Initialize the first two Fibonacci numbers
19     li $t1, 0 # F(0)
20     li $t2, 1 # F(1)
21
22     # Calculate the n-th Fibonacci number
23     beqz $t0, done # If n is 0, skip the loop
24
25 fib_loop:
26     add $t3, $t1, $t2 # F(n) = F(n-1) + F(n-2)
27     move $t1, $t2    # Shift values for next iteration
28     move $t2, $t3
29     addi $t0, $t0, -1 # Decrement n
30     bnez $t0, fib_loop
31
32     # Print the result
33     li $v0, 1
34     la $a0, output_result
35     syscall
```

```

32 done:
33     # Display the result
34     li $v0, 4
35     la $a0, output_result
36     syscall
37
38     # Display the n-th Fibonacci number (in $t1)
39     li $v0, 1
40     move $a0, $t1
41     syscall
42
43     # Exit the program
44     li $v0, 10
45     syscall

```

### Test & screenshot:

Clear

Enter a positive integer (n): 10  
The n-th Fibonacci number: 55  
— program is finished running —