Name:	Muhammad Attiq
Reg. No:	FA23–BCE–060
Section:	BCE–5A
Subject:	Computer Organization & Architecture
Instructor:	Dr. Irfanullah
Lab Report:	4

Lab # 04 Flow Control

Objectives

After completing this lab, you will:

- Get familiar with MIPS Jump and Branch instructions.
- Learn about pseudo instructions in MIPS.
- Learn how to translate high-level flow control constructs (if-then-else, for loop, while loop) to MIPS code.

In-Lab

Tasks

- 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.

```
.data
prompt: .asciiz"enter the integer: \n"
result: .asciiz"number of 1's binary representation: "

.text
main:
li $v0, 4
la $a0, prompt
syscall
li $v0, 5
syscall
move $t0, $v0
li $t2, 0

loop:
beq $t0, $zero, exit
andi $t1, $t0, 1
beq $t1, $zero, skip
addu $t2, $t2, 1

skip:
srl $t0, $t0, 1
j loop
```

```
24
25  exit:
26  li $v0, 4
27  la $a0, result
28  syscall
29  li $v0, 1
30  move $a0, $t2
31  syscall
32  li $v0, 10
33  syscall
```

```
enter the integer:
17
number of 1's binary representation: 2
```

Output

- 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.

```
1   .data
2   message_1: .asciiz"enter first integer: \n"
3   message_2: .asciiz"enter second integer: \n"
4   result: .asciiz"the result is: \n"
5
6   .text
7   main:
8   li $v0, 4
9   la $a0, message_1
10  syscall
11  li $v0, 5
12  syscall
13  move $t0, $v0
14  li $v0, 4
15  la $a0, message_2
16  syscall
17  li $v0, 5
18  syscall
19  move $t1, $v0
20  li $t2, 0
21
22  loop:
23  bgt $t0, $t1, exit
```

```
24  addu $t2, $t2, $t0
25  addi $t0, $t0, 1
26  j loop
27
28  exit:
29  li $v0, 4
30  la $a0, result
31  syscall
32  li $v0, 1
33  move $a0, $t2
34  syscall
35  li $v0, 10
36  syscall
```

```
enter first integer:
3
enter second integer:
7
the result is:25
```

Output

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

  The first two numbers are 0 and 1 and each subsequent number is the sum of the previous two. Write a program that asks the user to enter a positive integer number n and then prints the nth number in the Fibonacci sequence. The following algorithm can be used:

```
/* Input: n positive integer
   Output: nth Fibonacci number
fib0 = 0, fib1 = 1 */
for (i=2; i <= n; i++) do
        temp = fib0
        fib0 = fib1
        fib1 = temp + fib1
if (n > 0) fib = fib1
else fib = 0
```

```
.data
prompt:.asciiz"Enter the integer:\n"
result:.asciiz"The nth Fibonacci number is:\n"
.text
main:
li $v0,4
la $a0,prompt
syscall
li $v0,5
syscall
move $t0,$v0
beq $t0,$zero,fib_0
beq $t0,1,fib_1
li $t2,0
li $t3,1
li $t4,2
loop:
bgt $t4,$t0,fib_done
move $t5,$t2
move $t2,$t3
addu $t3,$t5,$t3
addi $t4,$t4,1
j loop
fib_0:
li $t6,0
j done
fib_1:
li $t6,1
j done
fib_done:
move $t6,$t3
done:
li $v0,4
la $a0,result
syscall
li $v0,1
move $a0,$t6
syscall
li $v0,10
syscall
```

```
Enter the integer:
5
The nth Fibonacci number is:
5
```

Output

Critical Analysis / Conclusion

Working on these tasks in MIPS assembly helped me see how a computer actually runs instructions. Counting 1's in a binary number improves understanding of bit operations, summing numbers between two values strengthens the use of loops and arithmetic, and the Fibonacci program shows how algorithms are built step by step with registers. Debugging these programs in MIPS also improves skills, as students can check register values, follow loops, and confirm results at each stage. Overall, these exercises connect theory with practice, build logical thinking, and prepare students for advanced topics in computer engineering.

Lab Assessment		
Pre Lab	**/5**	**/25**
Performance	**/5**	
Results	**/5**	
Viva	**/5**	
Critical Analysis	**/5**	
Instructor Signature and Comments		