Name:	Muhammad Attiq
Reg. No:	FA23 – BCE – 060
Section:	BCE – 5A
Subject:	Computer Organization & Architecture
Instructor:	Dr. Irfanullah
Lab Report:	3

Lab # 03 Integer Arithmetic

Objectives

After completing this lab, you will:

- Get familiar with the basic MIPS integer arithmetic and logic instructions including:
  - Integer addition and subtraction instructions
  - Bitwise logic instructions
  - Shift instructions
- Learn some useful applications of these instructions.

In-Lab

Tasks

- Write a program to ask the user to enter two integers A and B and then display the result of computing the expression: A + 2B - 5.

```
.data
prompt_for_A: .asciiz"Enter A: \n"
prompt_for_B: .asciiz"Enter B: \n"
result: .asciiz"Result: \n"

.text
main:
li $v0, 4
la $a0, prompt_for_A
syscall
li $v0, 5
syscall
move $t1, $v0 # $t1 = A
li $v0, 4
la $a0, prompt_for_B
syscall
li $v0, 5
syscall
move $t2, $v0 # $t2 = B

li $t0, -5
```

```
operation:

sll $t2, $t2, 1
add $t2, $t2, $t0
add $t1, $t1, $t2

li $v0, 4
la $a0, result
syscall

li $v0, 1
move $a0, $t1
syscall

li $v0, 10
syscall
```

```
Enter A:
3
Enter B:
4
Result: 6
```

Output

- 2. Assume that $s1 = 0x12345678 and $s2 = 0xffff9a00. Determine the content of registers $s3 to $s6 after executing the following instructions:

  and $s3, $s1, $s2 # $s3 =

  or $s4, $s1, $s2 # $s4 =

  xor $s5, $s1, $s2 # $s5 =

  nor $s6, $s1, $s2 # $s6 =

  Write a program to execute these instructions and verify the content of registers $s3 to $s6.

```
.data
and_result: .asciiz"AND RESULT: "
or_result: .asciiz"\nOR RESULT: "
xor_result: .asciiz"\nXOR RESULT: "
nor_result: .asciiz"\nNOR RESULT: "

.text
main:

li $s1, 0x12345678
li $s2, 0xffff9a00

and $s3, $s1, $s2
or $s4, $s1, $s2
xor $s5, $s1, $s2
nor $s6, $s1, $s2

li $v0, 4
la $a0, and_result
syscall
```

```
li $v0, 34
move $a0, $s3
syscall

li $v0, 4
la $a0, or_result
syscall

li $v0, 34
move $a0, $s4
syscall

li $v0, 4
la $a0, xor_result
syscall

li $v0, 34
move $a0, $s5
syscall

li $v0, 4
la $a0, nor_result
syscall

li $v0, 34
move $a0, $s6
syscall

li $v0, 10
syscall
```

```
AND RESULT: 0x12341200
OR RESULT: 0xffffde78
XOR RESULT: 0xedcbcc78
NOR RESULT: 0x00002187
```

Output

- Assume that $s1 = 0x87654321. Determine the content of registers $s2 to $s4 after executing the following instructions:

  sll $s2, $s1, 16 # $s2 =

  srl $s3, $s1, 8 # $s3 =

  sra $s4, $s1, 12 # $s4 =

  Write a program to execute these instructions and verify the content of registers $s2 to $s4.

```
.data
sll_result: .asciiz "SLL RESULT: "
srl_result: .asciiz "\nSRL RESULT: "
sra_result: .asciiz "\nSRA RESULT: "

.text
main:
    li $s1, 0x87654321

    sll $s2, $s1, 16
    srl $s3, $s1, 8
    sra $s4, $s1, 12

    li $v0, 4
    la $a0, sll_result
    syscall
    li $v0, 34
    move $a0, $s2
    syscall
    li $v0, 4
    la $a0, srl_result
    syscall
    li $v0, 34
    move $a0, $s3
    syscall

    li $v0, 4
    la $a0, sra_result
    syscall
    li $v0, 34
    move $a0, $s4
    syscall

    li $v0, 10
    syscall
```

```
SLL RESULT: 0x43210000
SRL RESULT: 0x00876543
SRA RESULT: 0xfff87654
```

Output

- Write a program that asks the user to enter an alphabetic character (either lower or upper case) and change the case of the character from lower to upper and from upper to lower and display it.

```
.data
prompt: .asciiz"enter alphabet \n"
result1: .asciiz "\nUppercase: "
result2: .asciiz "\nLowercase: "

.text
main:

li $v0, 4
la $a0, prompt
syscall

li $v0, 12
syscall

move $t0, $v0
move $s0, $t0
andi $s0, $s0, 0xDF

li $v0, 4
la $a0, result1
syscall

li $v0, 11
move $a0, $s0
syscall

move $s0, $t0

ori $s0, $s0, 0x20

li $v0, 4
la $a0, result2
syscall

li $v0, 11
move $a0, $s0
syscall
```

```
enter alphabet
r
Uppercase: R
Lowercase: r
```

Output

- Write a program that asks the user to enter and integer number and read it. Then ask him to enter a bit position (between 0 and 31) and display the value of that bit.

```
.data
prompt_num: .asciiz "Enter number: "
prompt_bit: .asciiz "Enter bit (0-31): "
result:     .asciiz "Bit value: "

.text
main:
    li $v0, 4
    la $a0, prompt_num
    syscall
    li $v0, 5
    syscall
    move $t0, $v0

    li $v0, 4
    la $a0, prompt_bit
    syscall
    li $v0, 5
    syscall
    move $t1, $v0

    srlv $t2, $t0, $t1
    andi $t3, $t2, 1
    li $v0, 4
    la $a0, result
    syscall
    li $v0, 1
    move $a0, $t3
    syscall

    li $v0, 10
    syscall
```

```
Enter number: 45
Enter bit (0-31): 4
Bit value: 0
```

Output

- Write a program that asks the user to enter a signed number and read it. Then display the content of multiplying this number by 24.

```
.data
prompt: .asciiz "Enter a number: "
result: .asciiz "Number * 24 = "
.text
main:
    li $v0, 4
    la $a0, prompt
    syscall
    li $v0, 5
    syscall
    move $t0, $v0

    sll $t1, $t0, 3
    sll $t2, $t0, 4
    add $t3, $t1, $t2
    li $v0, 4
    la $a0, result
    syscall
    li $v0, 1
    move $a0, $t3
    syscall
    li $v0, 10
    syscall
```

```
Enter a number: -2
Number * 24 = -48
```

Output

Critical Analysis

This lab improved my understanding of registers, input/output, shifting operations **sll, sllv, sra** and masking with **AND/OR**. At first it was confusing, but doing the tasks step by step made me better at debugging and logical coding.

Benefits

- **Input/Output:** I learned how to take values from the user and show results.
- **Math and Shifting:** I practiced using registers with add, sll, sllv, and sra.
- **Masking (AND/OR):** I understood how to clear or set specific bits in data.

- **Conditions and Loops:** I saw how decisions and repeats are done in assembly.
- **Memory Use:** I learned how values are stored and moved between registers and memory.
- **Debugging:** I became more careful, since even a small mistake can stop the code.

Conclusion

Overall, this lab gave me hands-on practice with shifting and masking along with other assembly basics, building a stronger base for computer architecture.

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