**Question1:** Implement a program (MIPS Assembly) which multiplies user input by 10 using only bit shift operations and addition. Check to see if your program is correct by using the *mult* and *mflo* operators. Your program should include a proper and useful prompt for input and print the results in a meaningful manner.

.data

prompt: .asciiz "Enter an integer: "

resultMsg: .asciiz "The result of multiplying by 10 using shift operations is: "

checkMsg: .asciiz "The result of multiplying by 10 using mult operator is: "

.text

main:

# Print the prompt

li $v0, 4

la $a0, prompt

syscall

# Read an integer from the user

li $v0, 5

syscall

move $t0, $v0 # Store input in $t0

# Multiply by 10 using shift operations

sll $t1, $t0, 3 # $t1 = $t0 \* 8

sll $t2, $t0, 1 # $t2 = $t0 \* 2

add $t3, $t1, $t2 # $t3 = $t1 + $t2 (effectively $t0 \* 10)

# Print the result of shift operations

li $v0, 4

la $a0, resultMsg

syscall

move $a0, $t3

li $v0, 1

syscall

# Verify using mult operator

li $t4, 10

mult $t0, $t4

mflo $t5 # Result in $t5

# Print the result of mult operation

li $v0, 4

la $a0, checkMsg

syscall

move $a0, $t5

li $v0, 1

syscall

# Exit program

li $v0, 10

syscall

Question 2: Write programs (MIPS Assembly) to evaluate the following expressions. The user should enter the variables, and the program should print back an answer. Prompt the user for all variables in the expression, and print the results in a meaningful manner. **The results should be as accurate as possible.**

a)

.data

prompt\_x: .asciiz "Enter value for x: "

prompt\_y: .asciiz "Enter value for y: "

prompt\_z: .asciiz "Enter value for z: "

resultMsg: .asciiz "The result of 5x + 3y + z is: "

.text

main:

# Input x, y, z

li $v0, 4

la $a0, prompt\_x

syscall

li $v0, 5

syscall

move $t0, $v0 # x in $t0

li $v0, 4

la $a0, prompt\_y

syscall

li $v0, 5

syscall

move $t1, $v0 # y in $t1

li $v0, 4

la $a0, prompt\_z

syscall

li $v0, 5

syscall

move $t2, $v0 # z in $t2

# Calculate 5x

sll $t3, $t0, 2 # $t3 = x \* 4

add $t3, $t3, $t0 # $t3 = x \* 5

# Calculate 3y

sll $t4, $t1, 1 # $t4 = y \* 2

add $t4, $t4, $t1 # $t4 = y \* 3

# Calculate 5x + 3y + z

add $t5, $t3, $t4 # $t5 = 5x + 3y

add $t6, $t5, $t2 # $t6 = 5x + 3y + z

# Print result

li $v0, 4

la $a0, resultMsg

syscall

move $a0, $t6

li $v0, 1

syscall

# Exit program

li $v0, 10

syscall

b)

.data

prompt\_x: .asciiz "Enter value for x: "

prompt\_y: .asciiz "Enter value for y: "

prompt\_z: .asciiz "Enter value for z: "

resultMsg: .asciiz "The result of ((5x + 3y + z) / 2) \* 3 is: "

.text

main:

# Input x, y, z (same as previous)

# ...

# Calculate 5x + 3y + z (same as previous)

# ...

# Divide by 2

sra $t7, $t6, 1 # $t7 = (5x + 3y + z) / 2

# Multiply by 3

sll $t8, $t7, 1 # $t8 = $t7 \* 2

add $t8, $t8, $t7 # $t8 = $t7 \* 3

# Print result

li $v0, 4

la $a0, resultMsg

syscall

move $a0, $t8

li $v0, 1

syscall

# Exit program

li $v0, 10

syscall

c)

.data

prompt\_x: .asciiz "Enter value for x: "

resultMsg: .asciiz "The result of x^3 + 2x^2 + 3x + 4 is: "

.text

main:

# Input x

li $v0, 4

la $a0, prompt\_x

syscall

li $v0, 5

syscall

move $t0, $v0 # x in $t0

# Calculate x^2 and x^3

mul $t1, $t0, $t0 # $t1 = x^2

mul $t2, $t1, $t0 # $t2 = x^3

# Calculate 2x^2

sll $t3, $t1, 1 # $t3 = 2x^2

# Calculate 3x

sll $t4, $t0, 1 # $t4 = 2x

add $t4, $t4, $t0 # $t4 = 3x

# Sum all terms

add $t5, $t2, $t3 # $t5 = x^3 + 2x^2

add $t6, $t5, $t4 # $t6 = x^3 + 2x^2 + 3x

addi $t7, $t6, 4 # $t7 = x^3 + 2x^2 + 3x + 4

# Print result

li $v0, 4

la $a0, resultMsg

syscall

move $a0, $t7

li $v0, 1

syscall

# Exit program

li $v0, 10

syscall

.data

prompt\_x: .asciiz "Enter value for x: "

prompt\_y: .asciiz "Enter value for y: "

resultMsg: .asciiz "The result of (4x / 3) \* y is: "

.text

main:

# Input x and y

li $v0, 4

la $a0, prompt\_x

syscall

li $v0, 5

syscall

move $t0, $v0 # x in $t0

li $v0, 4

la $a0, prompt\_y

syscall

li $v0, 5

syscall

move $t1, $v0 # y in $t1

# Calculate 4x

sll $t2, $t0, 2 # $t2 = 4x

# Divide by 3

li $t3, 3

div $t2, $t3

mflo $t4 # $t4 = 4x / 3

# Multiply by y

mul $t5, $t4, $t1 # $t5 = (4x / 3) \* y

# Print result

li $v0, 4

la $a0, resultMsg

syscall

move $a0, $t5

li $v0, 1

syscall

# Exit program

li $v0, 10

syscall

**Question3:** Write a program (MIPS Assembly) to retrieve two numbers from a user and swap those numbers using only the *XOR* operation. You should not use a temporary variable to store the numbers while swapping them. Your program should include a proper and useful prompt for input and print the results in a meaningful manner.

.data

prompt\_a: .asciiz "Enter the first number (a): "

prompt\_b: .asciiz "Enter the second number (b): "

resultMsg: .asciiz "After swapping, the numbers are:\n"

result\_a: .asciiz "a: "

result\_b: .asciiz "b: "

.text

main:

# Input a and b

li $v0, 4

la $a0, prompt\_a

syscall

li $v0, 5

syscall

move $t0, $v0 # a in $t0

li $v0, 4

la $a0, prompt\_b

syscall

li $v0, 5

syscall

move $t1, $v0 # b in $t1

# Swap using XOR

xor $t0, $t0, $t1 # a = a XOR b

xor $t1, $t0, $t1 # b = (a XOR b) XOR b = original a

xor $t0, $t0, $t1 # a = (a XOR b) XOR (original a) = original b

# Print results

li $v0, 4

la $a0, resultMsg

syscall

la $a0, result\_a

syscall

move $a0, $t0

li $v0, 1

syscall

li $v0, 11

li $a0, 10 # Newline

syscall

li $v0, 4

la $a0, result\_b

syscall

move $a0, $t1

li $v0, 1

syscall

# Exit program

li $v0, 10

syscall

**Question 4:** Using only *sll* and *srl*, implement a program to check if a user input value is even or odd. The result should print out *0* if the number is even or *1* if the number is odd. Your program (MIPS Assembly) should include a proper and useful prompt for input and print the results in a meaningful manner.

.data

prompt: .asciiz "Enter an integer: "

evenMsg: .asciiz "The number is even.\n"

oddMsg: .asciiz "The number is odd.\n"

resultMsg: .asciiz "Result (0 for even, 1 for odd): "

.text

main:

# Input number

li $v0, 4

la $a0, prompt

syscall

li $v0, 5

syscall

move $t0, $v0

# Isolate least significant bit

sll $t1, $t0, 31

srl $t1, $t1, 31 # $t1 now contains LSB

# Print result

li $v0, 4

la $a0, resultMsg

syscall

move $a0, $t1

li $v0, 1

syscall

# Print even/odd message

beqz $t1, print\_even

li $v0, 4

la $a0, oddMsg

syscall

j exit

print\_even:

li $v0, 4

la $a0, evenMsg

syscall

exit:

li $v0, 10

syscall

**Question 5:** Implement a program (MIPS Assembly) to prompt the user for two numbers, the first being any number and the second a prime number. Return to the user a *0* if the second number is a prime factor for the first one, otherwise any number if it is not. For example, if the user enters *60* and *5*, the program returns *0*. If the user enters *62* and *5*, the program returns *2*.

.data

prompt\_num1: .asciiz "Enter the first number: "

prompt\_prime: .asciiz "Enter the prime number: "

resultMsg: .asciiz "The result is: "

.text

main:

# Input first number and prime

li $v0, 4

la $a0, prompt\_num1

syscall

li $v0, 5

syscall

move $t0, $v0 # First number in $t0

li $v0, 4

la $a0, prompt\_prime

syscall

li $v0, 5

syscall

move $t1, $v0 # Prime in $t1

# Check if prime is a factor

div $t0, $t1

mfhi $t2 # Remainder in $t2

# Print result

li $v0, 4

la $a0, resultMsg

syscall

beqz $t2, is\_factor

# Not a factor, print remainder

move $a0, $t2

j print\_result

is\_factor:

# Is a factor, print 0

li $a0, 0

print\_result:

li $v0, 1

syscall

# Exit program

li $v0, 10

syscall