 **Northwestern Polytechnic University**

**EE488 - Computer Architecture**

**Homework Assignment #3**

**Due day: 10/31/2021**

**Instruction:**

1. **Push the answer sheet to GitHub in word file**
2. **Overdue homework submission could not be accepted.**
3. **Takes academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**
4. Implement a program (MIPS Assembly) which multiplies a 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 "\nEnter an integer: "

prompt1: .asciiz "\nResult by multiplying 10 is : "

.text

li $v0,4

la $a0,prompt #it will print prompt

syscall

li $v0,5

syscall #ask user input

move $t1,$v0 #save a to t1

sll $s0,$t1,3 #get 8x

sll $s1,$t1,1 #get 2x

add $s2,$s0,$s1 #get multiply by 10x

li $v0,4

la $a0,prompt1 #it will print prompt

syscall

move $a0,$s2

li $v0,1

syscall

1. 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.**

# Purpose: To calculate the result of 5\*x^ + 3\*y + z

.data

prompt1: .asciiz "Enter a value for x: "

prompt2: .asciiz "Enter a value for y: "

prompt3: .asciiz "Enter a value for z: "

result: .asciiz "The result is: "

.text

.globl main

main:

# Get input value, x

addi $v0, $zero, 4

la $a0, prompt1

syscall

addi $v0, $zero, 5

syscall

move $s0, $v0

# Get input value, y

addi $v0, $zero, 4

la $a0, prompt2

syscall

addi $v0, $zero, 5

syscall

move $s1, $v0

# Get input value, z

addi $v0, $zero, 4

la $a0, prompt3

syscall

addi $v0, $zero, 5

syscall

move $s2, $v0

# Calculate the result of 5\*x + 3\*y + z and store it in $s1.

mul $t0, $s0, 5

mul $t1, $s1, 3

add $t0, $t0, $t1

add $s1, $t0, $s2

# Print output

addi $v0, $zero, 4 # Print result string

la $a0, result

syscall

addi $v0, $zero, 1 # Print result

move $a0, $s1

syscall

#Exit program

addi $v0, $zero, 10

syscall

# Purpose: To calculate the result of ((5\*x^ + 3\*y + z)/2)\*3

.data

prompt1: .asciiz "Enter a value for x: "

prompt2: .asciiz "Enter a value for y: "

prompt3: .asciiz "Enter a value for z: "

result: .asciiz "The result is: "

.text

.globl main

main:

# Get input value, x

addi $v0, $zero, 4

la $a0, prompt1

syscall

addi $v0, $zero, 5

syscall

move $s0, $v0

# Get input value, y

addi $v0, $zero, 4

la $a0, prompt2

syscall

addi $v0, $zero, 5

syscall

move $s1, $v0

# Get input value, z

addi $v0, $zero, 4

la $a0, prompt3

syscall

addi $v0, $zero, 5

syscall

move $s2, $v0

# Calculate the result of ((5\*x + 3\*y + z)/2)\*3 and store it in $s1.

mul $t0, $s0, 5

mul $t1, $s1, 3

add $t0, $t0, $t1

add $t0, $t0, $s2

div $t0, $t0, 2

mul $s1, $t0, 3

# Print output

addi $v0, $zero, 4 # Print result string

la $a0, result

syscall

addi $v0, $zero, 1 # Print result

move $a0, $s1

syscall

#Exit program

addi $v0, $zero, 10

syscall



# Purpose: To calculate the result of x\*x\*x+2\*x\*x+3x+4

.data

prompt: .asciiz "Enter a value for x: "

result: .asciiz "The result is: "

.text

.globl main

main:

# Get input value, x

addi $v0, $zero, 4

la $a0, prompt

syscall

addi $v0, $zero, 5

syscall

move $s0, $v0

# Calculate the result of x\*x\*x+2\*x\*x+3x+4 and store it in $s1.

mul $t0, $s0, $s0

mul $t1, $t0, $s0

mul $t0, $t0,2

add $t0, $t1, $t0

mul $t1, $s0, 3

add $t0, $t0, $t1

addi $s1, $t0, 4

# Print output

addi $v0, $zero, 4 # Print result string

la $a0, result

syscall

addi $v0, $zero, 1 # Print result

move $a0, $s1

syscall

#Exit program

addi $v0, $zero, 10

syscall

# Purpose: To calculate the result of ((4\*x)/3)\*y

.data

prompt1: .asciiz "Enter a value for x: "

prompt2: .asciiz "Enter a value for y: "

result: .asciiz "The result is: "

.text

.globl main

main:

# Get input value, x

addi $v0, $zero, 4

la $a0, prompt1

syscall

addi $v0, $zero, 5

syscall

move $s0, $v0

# Get input value, y

addi $v0, $zero, 4

la $a0, prompt2

syscall

addi $v0, $zero, 5

syscall

move $s1, $v0

# Calculate the result of ((4\*x)/3)\*y and store it in $s1.

mul $t0, $s0, 4

div $t0, $t0,3

mul $s1, $t0, $s1

# Print output

addi $v0, $zero, 4 # Print result string

la $a0, result

syscall

addi $v0, $zero, 1 # Print result

move $a0, $s1

syscall

#Exit program

addi $v0, $zero, 10

syscall

1. 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

prompt1: .asciiz "Enter a value for x: "

prompt2: .asciiz "Enter a value for y: "

result1: .asciiz "The value of x is: "

result2: .asciiz "\nThe value of y is: "

.text

.globl main

main:

# Get input value, x

addi $v0, $zero, 4

la $a0, prompt1

syscall

addi $v0, $zero, 5

syscall

move $s0, $v0

# Get input value, y

addi $v0, $zero, 4

la $a0, prompt2

syscall

addi $v0, $zero, 5

syscall

move $s1, $v0

xor $s0, $s0, $s1

xor $s1, $s0, $s1

xor $s0, $s0, $s1

# Print output

addi $v0, $zero, 4 # Print result string

la $a0, result1

syscall

addi $v0, $zero, 1 # Print result

move $a0, $s0

syscall

addi $v0, $zero, 4 # Print result string

la $a0, result2

syscall

addi $v0, $zero, 1 # Print result

move $a0, $s1

syscall

#Exit program

addi $v0, $zero, 10

syscall

1. 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 a number to be check if even or odd: "

message: .asciiz "\nResult"

msgEven: .asciiz "\n0\nNumber is even"

msgOdd: .asciiz "\n1\nNumber is odd"

.text

main:

#prompt for input

li $v0,4

la $a0,prompt

syscall

li $v0,5

syscall

move $t0,$v0

srl $s0,$t0,1 # Shift right by 1 bit

sll $t1,$s0,1 # shift left by 1 bit

#if number is even then original number is equal to the number after SLL and SRL

beq $t0,$t1, resultEven

bne $t0,$t1, resultOdd

#END

li $v0,10

la $a0,message

syscall

resultEven: # Jump here when number is even

li $v0,4

la $a0,msgEven

syscall

li $v0,10

la $a0,message

syscall

resultOdd: # jump here when number is odd

li $v0,4

la $a0,msgOdd

syscall

li $v0,10

la $a0,message

syscall

1. 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:.asciiz "Enter n integer: "

prompt1:.asciiz "Enter a prime number:"

message: .asciiz "\nResult"

msgfactor: .asciiz "\n0\Prime Factor"

msgnotfactor: .asciiz "\n1\nNot a Prime Factor"

.text

main:

#prompt for input

li $v0,4

la $a0,prompt

syscall

li $v0,5

syscall

move $t0,$v0

li $v0,4

la $a0,prompt1

syscall

li $v0,5

syscall

move $t1,$v0

div $t0,$t0,$t1

mfhi $t1

beq $t1,$zero, factor

bne $t1,$zero,notfactor

#END

li $v0,10

la $a0,message

syscall

factor:

li $v0,4

la $a0,msgfactor

syscall

li $v0,10

la $a0,message

syscall

notfactor:

li $v0,4

la $a0,msgnotfactor

syscall

li $v0,10

la $a0,message

syscall