**LABSHEET 5**

1. Write a program that reads two integer numbers A and B. The program must to indicate if one of these numbers is multiple of the other one

Hint: mfhi $t0 means move remainder to $t0 and mflo $t0 means move quotient to register $t0

Both are to be used after div instruction.

bge $t1,$t2,label

.data

MSG1: .asciiz "Enter number 1: "

MSG2: .asciiz "Enter number 2: "

MSG3: .asciiz "Number 1 "

MSG4: .asciiz "Number 2 "

MSG5: .asciiz "is divisible by "

MSG6: .asciiz "is not divisible by "

MSG7: .asciiz "\n"

.text

main:

    li $v0, 4

    la $a0, MSG1

    syscall

    li $v0, 5

    syscall

    move $t0, $v0

    li $v0, 4

    la $a0, MSG2

    syscall

    li $v0, 5

    syscall

    move $t1, $v0

    div $t0, $t1

    mfhi $t2

    beq $t2, 0, abc

    li $v0, 4

    la $a0, MSG3

    syscall

    li $v0, 4

    la $a0, MSG6

    syscall

    li $v0, 4

    la $a0, MSG4

    syscall

    li $v0, 4

    la $a0, MSG7

    syscall

    j next

abc:

    li $v0, 4

    la $a0, MSG3

    syscall

    li $v0, 4

    la $a0, MSG5

    syscall

    li $v0, 4

    la $a0, MSG4

    syscall

    li $v0, 4

    la $a0, MSG7

    syscall

    j exit

next:

    div $t1, $t0

    mfhi $t2

    beq $t2, 0, def

    li $v0, 4

    la $a0, MSG4

    syscall

    li $v0, 4

    la $a0, MSG6

    syscall

    li $v0, 4

    la $a0, MSG3

    syscall

    li $v0, 4

    la $a0, MSG7

    syscall

    j exit

def:

    li $v0, 4

    la $a0, MSG4

    syscall

    li $v0, 4

    la $a0, MSG5

    syscall

    li $v0, 4

    la $a0, MSG3

    syscall

    li $v0, 4

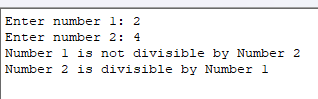
    la $a0, MSG7

    syscall

    j exit

exit:

    jr $ra



1. Write the complete assembly language program, including data declarations, that corresponds to the following C code fragment. Make use of the fact that multiplication and division by powers of 2 can be performed most efficiently by shifting

int main()

{ int K, Y ;

int Z ;

Y = 60 ;

K = 20;

Z = Y - 16 \* ( K / 4 + 210) ;

}

.data

    MSG1: .asciiz "Y: "

    MSG2: .asciiz "K: "

    MSG3: .asciiz "Z: "

    newline: .asciiz "\n"

.text

main:

    li $t1, 60 # y=60

    li $t2, 20 # k=20

    srl $t0, $t2, 2 # k/4

    addi $t0, $t0, 210 # k/4+210

    sll $t0, $t0, 4 # (k/4+210)\*16

    sub $t0, $t1, $t0 # y-(k/4+210)\*16

    li $v0, 4

    la $a0, MSG1

    syscall

    li $v0, 1

    move $a0, $t1

    syscall

    li $v0, 4

    la $a0, newline

    syscall

    li $v0, 4

    la $a0, MSG2

    syscall

    li $v0, 1

    move $a0, $t2

    syscall

    li $v0, 4

    la $a0, newline

    syscall

    li $v0, 4

    la $a0, MSG3

    syscall

    li $v0, 1

    move $a0, $t0

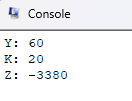
    syscall

    li $v0, 4

    la $a0, newline

    syscall

    jr $ra



1. Write down the MIPS code for the following C code and test the output

1. **If … else**

if ($t10 < 0) then

{

$s0 = 0 - $t8;

$t1 = $t1 +2;

}

else

{

$s0 = $t8;

$t2 = $t2 + 1;

}

IF:

    bge     $t10, $zero, ESLE

    sub     $s0, $zero, $t8

    addi    $t1, $t1, 2

    j       END

ESLE:

    move   $s0, $t8

    addi   $t2, $t2, 1

END:

    jr $ra

1. **For loop**

$t2 = 0;

for ( $t0 =8; $t0 > 0; $t0 = $t0 -1) do {$t2 = $t2 + $t0}

    li $t2, 0

    li $t0, 8

LOOP:

    bge $t0, $zero, END

    add $t2, $t2, $t0

    j LOOP

END:

    jr $ra