**Ejercicios RV32I - Aritmética entera**

1º Desarrolla un programa en ensamblador RV32I capaz de calcular el registro temporal (t0-t6) cuyo valor sea más alto, guardando dicho valor en el registro s0.

2º Desarrolla un programa en ensamblador RV32I que calcule la suma de todos los números positivos menores que el valor almacenado en el registro t0. En caso de que el registro contenga un valor negativo el resultado, obviamente, será 0.

3º Desarrolla un programa en ensamblador RV32I que calcule el “valor absoluto” del valor almacenado en el registro t0.

4º Desarrolla un programa en ensamblador RV32I que calcule el cuadrado el valor (codificado en binario puro ***sin signo***) almacenado en el registro t0. Repite el ejercicio considerando que el valor almacenado en t0 está codificado en binario puro con signo en complemento a 2.

5º Desarrolla un programa en ensamblador RV32I que active el registro t0 cuando el valor almacenado en el registro t1 es primo. Si es un número divisible, t0 quedará con el valor 0.

**Solución 1º**

.text

add s0, t0, zero #Comenzamos cargando t0 en s0

bge s0, t1, mirart2 #Comparamos t0 con t1

add s0, t1, zero #si t1>=s0 (t0)) restauramos s0

mirart2: bge s0, t2, mirart3 #Ahora miramos t2

add s0, t2, zero #si t2>=a0 restauramos s0

mirart3: bge s0, t3, mirart3 #Ahora miramos t3

add s0, t3, zero #si t3>=a0 restauramos s0

mirart4: bge s0, t4, mirart3 #Ahora miramos t4

add s0, t4, zero #si t4>=a0 restauramos s0

mirart5: bge s0, t5, mirart3 #Ahora miramos t5

add s0, t5, zero #si t5>=a0 restauramos s0

mirart6: bge s0, t6, fin #Ahora miramos t6

add s0, t6, zero #si t6>=a0 restauramos s0

fin: ebreak

**Solución 2º**

.text

#Es importante comprobar si el número almacenado en t0 es positivo

add t1, zero, zero #Resultado en t1

add t2, zero, zero #contador en t2

ble t0, zero, fin #si negativo fin

loop: bge t2, t0, fin #si contador > valor fin

add t1, t1, t2 #resultado += contador

addi t2, t2, 1 #contador ++

beq zero, zero, loop #salto incondicional a loop

fin: ebreak

**Solución 3º**

.text

#Hay que evaluar el signo y si es negativo complementar a 2

bge t0, zero, fin #si es positivo o cero ya está

xori t0, t0, -1 #si negativo, primero C1

addi t0, t0, 1 # luego +1 (así C2)

fin: ebreak

**Solución 4º**

.text

add t1, zero, zero #acumulador = 0

add t2, t0, zero #contador = Valor

loop: beq t2, zero fin #si contador==0 fin

add t1, t0, t1 #acumulador+=Valor

addi t2, t2, -1 #contador--

beq zero, zero, loop #saltar a bucle

fin: ebreak

**Solución 5º**

.text

ori t0, zero, 2557 #t0 valor a comprobar

srli t1, t0, 1 #t1 = t0/2 (máx divisor posible)

loop: slti t6, t1, 2 #t6 set si t1 < 2

bne t6, zero, siprimo #es primo porque no ha salido

add t3, t0, zero #t3 número a dividir

add t2, t1, zero #t2 siguiente posible valor

jal ra, division #realiza t3/t2

beq t5, zero, noprimo #si resto es 0 no es primo

addi t1, t1, -1 #decrementar posible divisor

beq zero, zero, loop #volver a probar

siprimo: addi t0, zero, 1

beq zero, zero, fin

noprimo: add t0, zero, zero

fin: ebreak

# Realiza la división de t3(DDO))/t2(DOR)) y almacena en el #registro t4 el valor del cociente y en t5 el valor del resto

division: add t4, zero, zero #Cociente = 0

add t5, t3, zero #Resto = DOR

loopDiv: blt t3, t2, fin #Si el DDO < DOR fin

sub t3, t3, t2 #DDO = DDO - DOR

addi t4, t4, 1 #Cociente +=1

beq zero, zero, loopDiv

fin add t5, t3, zero #resto = DOR (lo que queda)

jalr t6, 0(ra)