**Báo cáo Thực hành KTMT buổi 3**

**Họ và tên:** Nguyễn Đức Phú

**MSSV:** 20215116

**Assignment 1:**

Graphical user interface, text, application

Description automatically generated

* **Khởi tạo các giá trị x, y, z lần lượt là 2, 7, 3**
* Sau khi chạy các câu lệnh:addi $s1, $zero, 7  
  addi $s2, $zero, 4
* Các thanh ghi $s1, $s2 thay đổi thành các giá trị 7, 4
* Sau khi chạy các câu lệnh:

la $a0, x  
lw $t1, 0($a0)  
la $a0, y  
lw $t2, 0($a0)  
la $a0, z  
lw $t3, 0($a0)

* Các lệnh la đã lấy địa chỉ ô nhớ của biến và lưu vào thanh ghi tạm $at, lệnh lw để ghi giá trị đó vào các thanh ghi $t1, $t2, $t3
* Sau khi chạy câu lệnh:

slt $t0,$s2,$s1

* Câu lệnh này sẽ thực hiện so sánh $s2 < $s1 (tức j < i), nếu  
  đúng thì thanh ghi $t0 được gán giá trị bằng 1, nếu sai thanh ghi $t0 được gán giá trị bằng 0.
* Sau khi chạy xong câu lệnh ta được kết quả $t0 = 1 do $s2 < $s1 (4 < 7) là đúng
* Sau khi chạy câu lệnh:

bne $t0,$zero,else

* Câu lệnh so sánh thanh ghi $t0 với thanh ghi $zero, nếu không  
  bằng nhau thì sẽ nhảy đến label else
* Thanh ghi pc lúc này từ giá trị 0x00400030 nhảy đến else có giá trị 0x00400040 vì $t0 =0x00000001 khác $zero
* Sau khi chạy các câu lệnh:

addi $t2,$t2,-1  
add $t3,$t3,$t3

* Hai câu lệnh trên thực hiện hai phép toán y = y - 1 và z = 2\*z
* Thanh ghi $t2 = 0x00000006 (7-1=6)

Và $t3 = 0x00000006 (2\*3=6)

* Kết quả đúng
* Sau khi chạy câu lệnh:

endif

* Kết thúc chương trình
* Nếu thay đổi giá trị khởi tạo i, j sao cho i <= j thì giá trị $t0 = 0
* Tại lệnh: bne $t0,$zero,else => thanh ghi pc sẽ không thảy tới nhãn else mà chỉ tăng thêm 4 và tiếp tục thực hiện các lệnh

addi $t1,$t1,1  
addi $t3,$zero,1

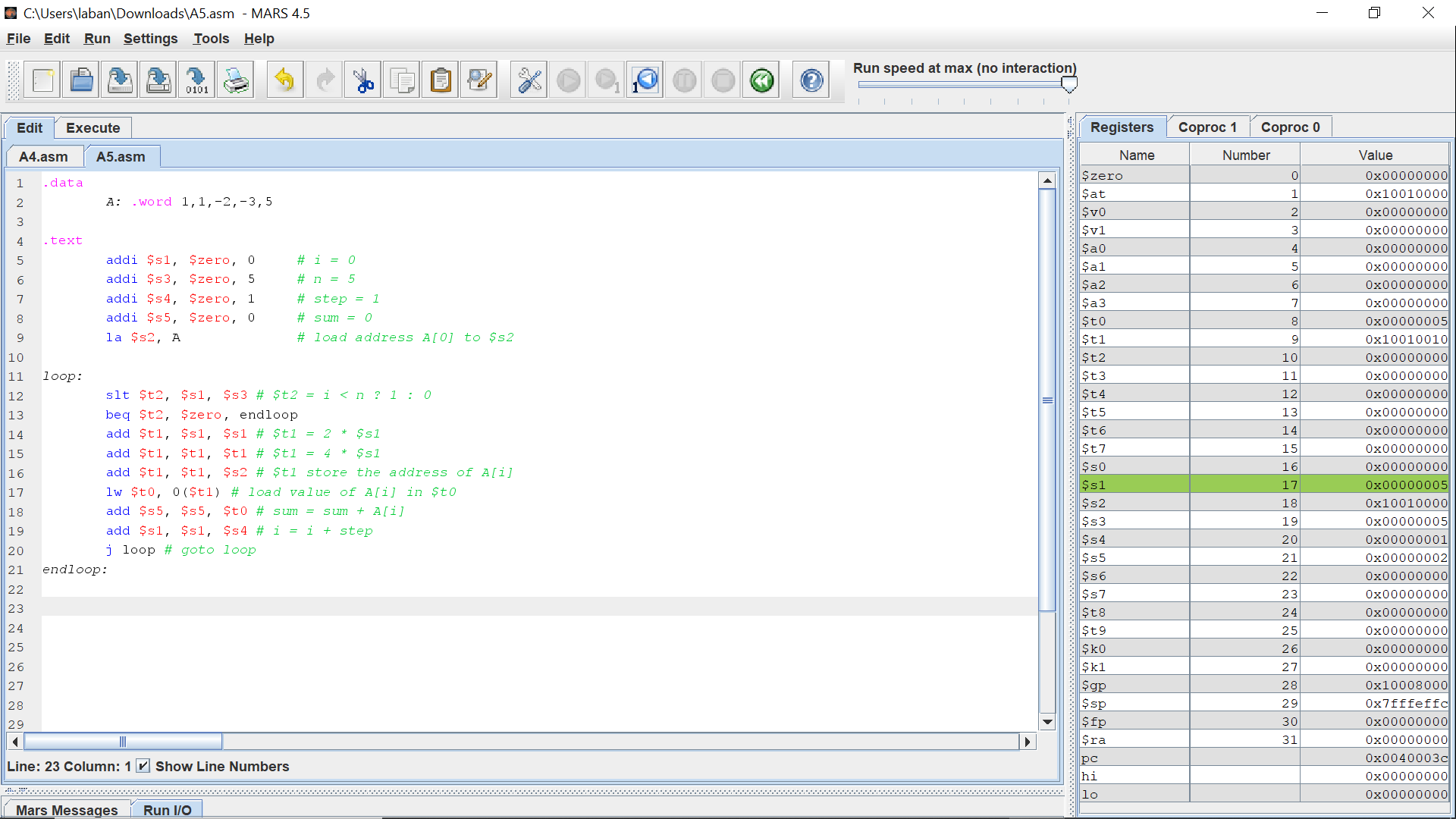
* Thực hiện 2 phép toán x=x+1 và z=1
* Thanh ghi $t1 = 0x00000003 (2+1=3)

Và $t3 = 0x00000001 ( =1)

* Tới dòng lệnh j endif thì thanh ghi pc nhảy tới nhãn endif (0x00400048) và kết thúc chương trình

**Assignment 2:**

* ***Khởi tạo mảng A gồm 5 phần tử {1,1,-2,-3,5} và i = 0, n =5, step = 1, sum = 0***



* Sau khi chạy các câu lệnh:

addi $s1, $zero, 0  
addi $s3, $zero, 5  
addi $s4, $zero, 1  
addi $s5, $zero, 0

* Các thanh ghi $s1, $s3, $s4, $s5 được gán giá trị lần lượt tương đương với i=0, n=5, step=1, sum=0
* Sau khi chạy câu lệnh:

la $s2, A

* Địa chỉ của mảng A được nạp vào thanh ghi $s2, đây cũng chính là địa chỉ cơ sở A[0]
* Sau khi chạy câu lệnh:

slt $t2, $s1, $s3

* Câu lệnh kiểm tra i < n không. Vì $s1 < $s3 nên $t2 = 1
* **Sau khi chạy câu lệnh:**

beq $t2, $zero, endloop

* Vì $t2 khác $zero nên thanh ghi pc trỏ đến địa chỉ của câu lệnh tiếp theo mà không nhảy đến nhãn *endloop*
* **Sau khi chạy các câu lệnh:**

add $t1, $s1, $s1  
add $t1, $t1, $t1  
add $t1, $t1, $s2

* Thực hiện các phép toán khiến $t1 = 4\*i và cộng $t1với $s2 (địa chỉ cơ sở A[0])
* Có được địa chỉ A[i]
* **Sau khi chạy lệnh:**

lw $t0, 0($t1)

* Câu lệnh lấy dữ liệu trong ô nhớ có địa chỉ $t1 và nạp vào $t0
* Sau khi chạy các câu lệnh:

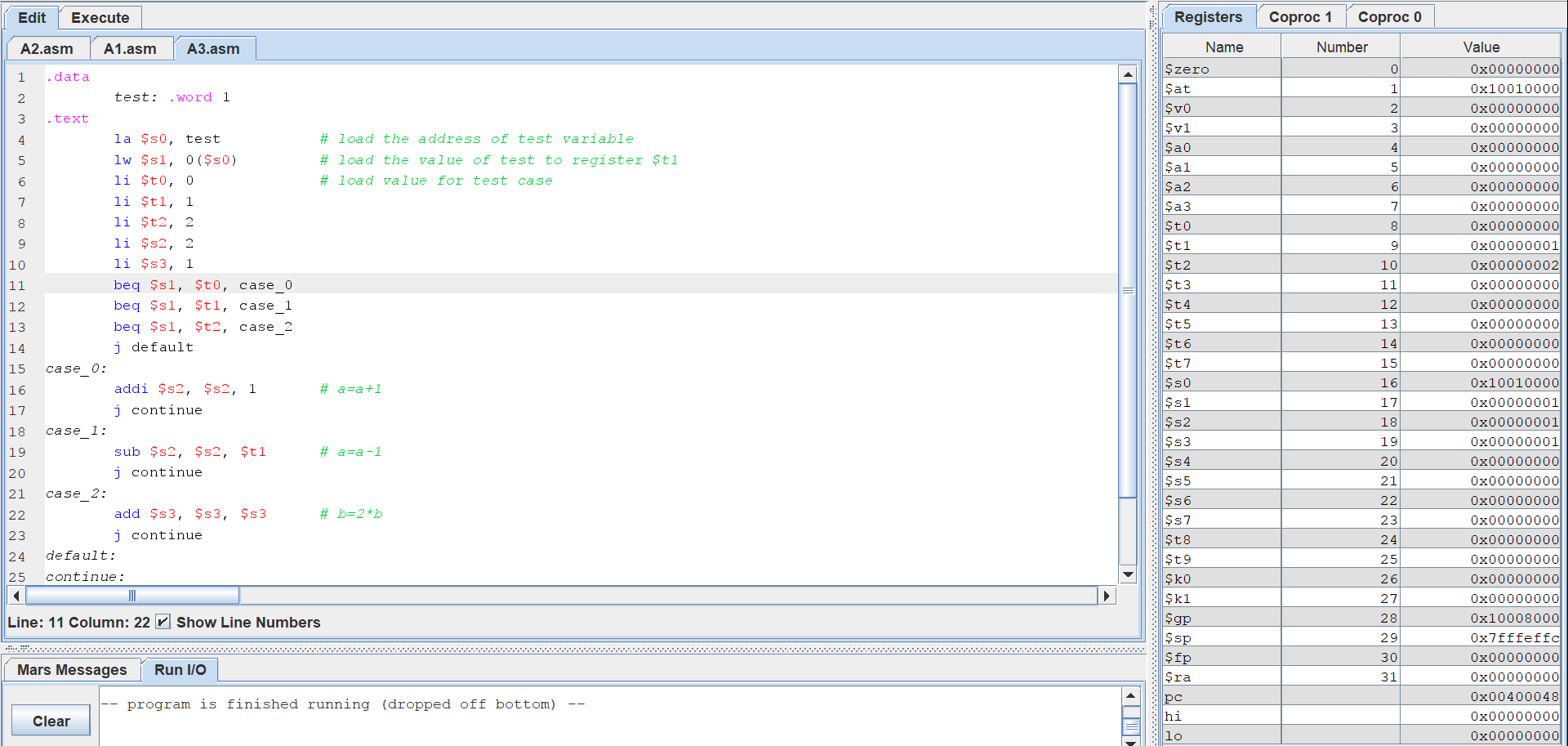
add $s5, $s5, $t0  
add $s1, $s1, $s4

* Hai câu lệnh thực hiện tính sum = sum + A[i] và i = i + step
* **Sau khi chạy lệnh:**

j loop

* Câu lệnh nhảy đến nhãn loop
* Khi i = 5, và câu lệnh beq có $t2 = $zero, câu lệnh tiếp theo nhảy đến nhãn *endloop* và kết thúc chương trình
* Kết quả sum = $s5 =0x00000002 là kết quả đúng của **1+1+(-2)+(-3)+5**

**Assignment 3:**



Dữ liệu của *test* được ghi vào $s1, so sánh $s1 với $t0, $t1, $t2  
$s1 = $t0 => nhảy tới *case\_0*$s1 = $t1 => nhảy tới case\_1  
$s1 = $t2 => nhảy tới *case\_2*Khi cả ba đều không bằng $s1 khi chạy đến lệnh j default thì lệnh sẽ nhảy đến nhãn *default*Trường hợp này $s1 = $t1 = 1 nên sẽ thực hiện lệnh trong *case\_1*

* Kết quả thu được: $s2 = $s2 – 1 = 2 – 1 = 1

Với trường hợp $s1 = $t0 = 0 => Thực hiện *case\_0*

* Kết quả: $s2 = $s2 + 1 = 2 + 1 = 3

Với trường hợp $s1 = $t2 = 2 => Thực hiện *case\_2*

* Kết quả: $s3 = $s3 + $s3 = 1 + 1 = 2
* Sau khi thực hiện các case thì tới lệnh j continue, nhảy tới nhãn continue và kết thúc chương trình

Với trường hợp $s1 không bằng giá trị nào trong 3 giá trị $t0, $t1, $t2

* Không nhảy tới case nào, pc tiếp tục tăng 4 tới j default nhảy tới nhãn default, tiếp tục tăng 4 tới continue và kết thúc chương trình

**Assignment 4:**

1. i < j

.data

x: .word 2

y: .word 7

z: .word 3

.text

addi $s1, $zero, 7

addi $s2, $zero, 4

la $a0, x

lw $t1, 0($a0)

la $a0, y

lw $t2, 0($a0)

la $a0, z

lw $t3, 0($a0)

start:

**slt $t0,$s1,$s2 # i<j**

**beq $t0,$zero,else # branch to else if i>=j**

addi $t1,$t1,1 # then part: x=x+1

addi $t3,$zero,1 # z=1

j endif # skip “else” part

else:

addi $t2,$t2,-1 # begin else part: y=y-1

add $t3,$t3,$t3 # z=2\*z

endif:

1. i >= j

.data

x: .word 2

y: .word 7

z: .word 3

.text

addi $s1, $zero, 7

addi $s2, $zero, 4

la $a0, x

lw $t1, 0($a0)

la $a0, y

lw $t2, 0($a0)

la $a0, z

lw $t3, 0($a0)

start:

**slt $t0,$s1,$s2 # i<j**

**bne $t0,$zero,else # branch to else if i<j**

addi $t1,$t1,1 # then part: x=x+1

addi $t3,$zero,1 # z=1

j endif # skip “else” part

else:

addi $t2,$t2,-1 # begin else part: y=y-1

add $t3,$t3,$t3 # z=2\*z

endif:

1. i + j <= 0

.data

x: .word 2

y: .word 7

z: .word 3

.text

addi $s1, $zero, 7

addi $s2, $zero, 4

la $a0, x

lw $t1, 0($a0)

la $a0, y

lw $t2, 0($a0)

la $a0, z

lw $t3, 0($a0)

start:

**add $t4, $s1,$s2 # $t4=i+j**

**sgt $t0,$t4,0 # $t4 > 0**

**bne $t0,$zero,else # branch to else if i+j >0**

addi $t1,$t1,1 # then part: x=x+1

addi $t3,$zero,1 # z=1

j endif # skip “else” part

else:

addi $t2,$t2,-1 # begin else part: y=y-1

add $t3,$t3,$t3 # z=2\*z

endif:

1. i + j = m + n

.data

x: .word 2

y: .word 7

z: .word 3

.text

addi $s1, $zero, 7

addi $s2, $zero, 4

la $a0, x

lw $t1, 0($a0)

la $a0, y

lw $t2, 0($a0)

la $a0, z

lw $t3, 0($a0)

**addi $s5,$zero,1 #m=1**

**addi $s6,$zero,2 #n=2**

start:

**add $t4, $s1,$s2 # $t4=i+j**

**add $t5, $s5,$s6 # $t5=m+n**

**slt $t0,$t5,$t4 # $t5 < $t4**

**beq $t0,$zero,else # go to else if m+n >= i+j**

addi $t1,$t1,1 # then part: x=x+1

addi $t3,$zero,1 # z=1

j endif # skip “else” part

else:

addi $t2,$t2,-1 # begin else part: y=y-1

add $t3,$t3,$t3 # z=2\*z

endif:

**Assignment 5:**

1. i < n (đã làm)
2. i <= n

.data

A: .word 1,1,-2,-3,5

.text

addi $s1, $zero, 0 # i = 0

addi $s3, $zero, 5 # n = 5

addi $s4, $zero, 1 # step = 1

addi $s5, $zero, 0 # sum = 0

la $s2, A # load address A[0] to $s2

loop:

**slt $t2, $s3, $s1 # $t2 = n < i ? 1 : 0**

**bne $t2, $zero, endloop # go to end if i>n**

add $t1, $s1, $s1 # $t1 = 2 \* $s1

add $t1, $t1, $t1 # $t1 = 4 \* $s1

add $t1, $t1, $

lw $t0, 0($t1)

add $s5, $s5, $t0 # sum = sum + A[i]

add $s1, $s1, $s4 # i = i + step

j loop # goto loop

endloop:

1. sum >= 0

.data

A: .word 1,1,-2,-3,5

.text

addi $s1, $zero, 0 # i = 0

addi $s3, $zero, 5 # n = 5

addi $s4, $zero, 1 # step = 1

addi $s5, $zero, 0 # sum = 0

la $s2, A # load address A[0] to $s2

loop:

**slt $t2, $s5, $zero # $t2 = sum < 0 ? 1 : 0**

**bne $t2, $zero, endloop # go to end if sum <0**

add $t1, $s1, $s1 # $t1 = 2 \* $s1

add $t1, $t1, $t1 # $t1 = 4 \* $s1

add $t1, $t1, $s2

lw $t0, 0($t1)

add $s5, $s5, $t0 # sum = sum + A[i]

add $s1, $s1, $s4 # i = i + step

j loop # goto loop

endloop:

1. A[i] == 0

.data

A: .word 1,1,-2,0,5

.text

addi $s1, $zero, 0 # i = 0

addi $s3, $zero, 5 # n = 5

addi $s4, $zero, 1 # step = 1

addi $s5, $zero, 0 # sum = 0

la $s2, A # load address A[0] to $s2

loop:

slt $t2, $s1, $s3 # $t2 = i < n ? 1 : 0

beq $t2, $zero, endloop

add $t1, $s1, $s1 # $t1 = 2 \* $s1

add $t1, $t1, $t1 # $t1 = 4 \* $s1

add $t1, $t1, $s2 # $t1 store the address of A[i]

lw $t0, 0($t1) **# $t0=A[i]**

**beq $t0,$zero, endloop #if A[i]=0 => end**

add $s5, $s5, $t0 # sum = sum + A[i]

add $s1, $s1, $s4 # i = i + step

j loop # goto loop

endloop:

**Assignment 6:**

.data

A: .word 1, -3, -10, 6, -29, 3, -39

message: .asciiz "Tri tuyet doi lon nhat la: "

.text

addi $s0, $zero, 0 # max = 0

la $a0, A

lw $s1, 0($a0) # A[0]

addi $s2, $zero, 0 # i = 0

addi $s3, $zero, 7 # n = 7

loop: slt $t2, $s2, $s3 # i<n

beq $t2, $zero, endloop # i>=n branch to endloop

sll $t1, $s2, 2 # t1 = i \* 4

add $a1, $a0, $t1 # a1 = a0 + 4

lw $s4, 0($a1) # s4 = A[i]

if\_nhohon\_0:

bgez $s4, if\_lonhon\_0 # s4>0 branch to if\_lonhon

sub $s4, $zero, $s4 # s4 = 0 - s4

j if\_lonhon\_0

if\_lonhon\_0:

slt $t4, $s0, $s4 # max < s4

bne $t4, $zero, max # max < s4 branch to max

j reloop # jump reloop

max: add $s0, $zero, $s4 # max = 0 + s4

j reloop # jump reloop

reloop:

addi $s2, $s2, 1 # i=i+1

j loop # jump loop

endloop: