Laboratory Exercise 3 – Report:

Load/ Store , Jump & Branch instructions

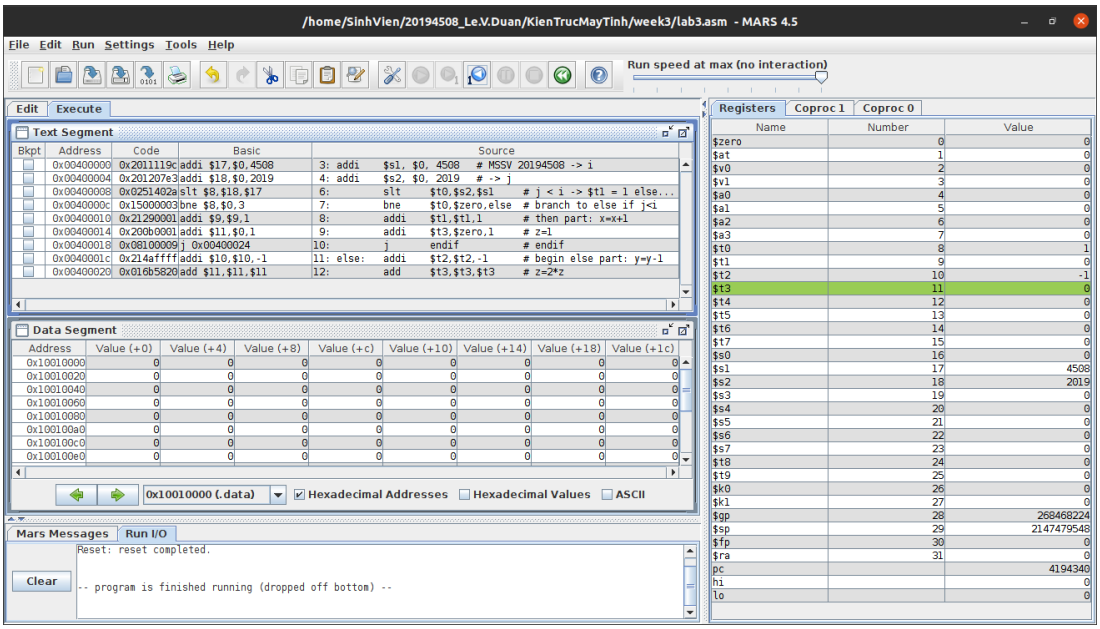
Lê Văn Duẩn - 20194508

1. Assignment 1

- Mã nguồn:

```
#Laboratory Exercise 3, Home Assignment 1
.text
addi    $s1, $0, 4508          # MSSV 20194508 -> i
addi    $s2, $0, 2019          # -> j
start:
slt     $t0,$s2,$s1  # j < i -> $t0 = 1 else $t0 = 0
bne     $t0,$zero,else         # branch to else if j<i
addi    $t1,$t1,1      # then part: x=x+1
addi    $t3,$zero,1  # z=1
j       endif          # endif
else:   addi    $t2,$t2,-1     # begin else part: y=y-1
add     $t3,$t3,$t3  # z=2*z
endif:
```

- Kết quả chạy mô phỏng:



- Giải thích sự thay đổi của các thanh ghi:

+ Sau 2 lệnh đầu tiên gán giá trị cho các thanh ghi $s1 và $s2 lần lượt là 4508 và 2019 -> sau 2 lệnh giá trị của 2 thanh ghi đó thay đổi từ 0 sang giá trị tương ứng.

+ Sau khi thực thi lệnh slt $t0,$s2,$s1 -> so sánh giá trị các thanh ghi $s2 và $s1 và trả kết quả vào thanh ghi $t0. Vì $s2 có giá trị lớn hơn giá trị trong thanh ghi $s1 (4508 > 2019) -> $t0 = 1

+ lệnh tiếp theo: bne $t0,$zero,else -> so sánh giá trị $t0 với $0 nếu đúng thì chạy các lệnh tiếp theo nếu không thì đi đến lệnh nhãn else -> trong TH này vì $t0 = 1 khác $0 nên sẽ rẽ nhánh đến nhãn else

+ Vì giá trị của thanh ghi $t2 và $t3 ban đầu đều bằng 0 nên sau khi thực thi 2 lệnh ở nhãn else thì giá trị của nó sẽ là $t2 = -1, $t3 = 0 -> đúng như t thay đổi trong chạy mô phỏng

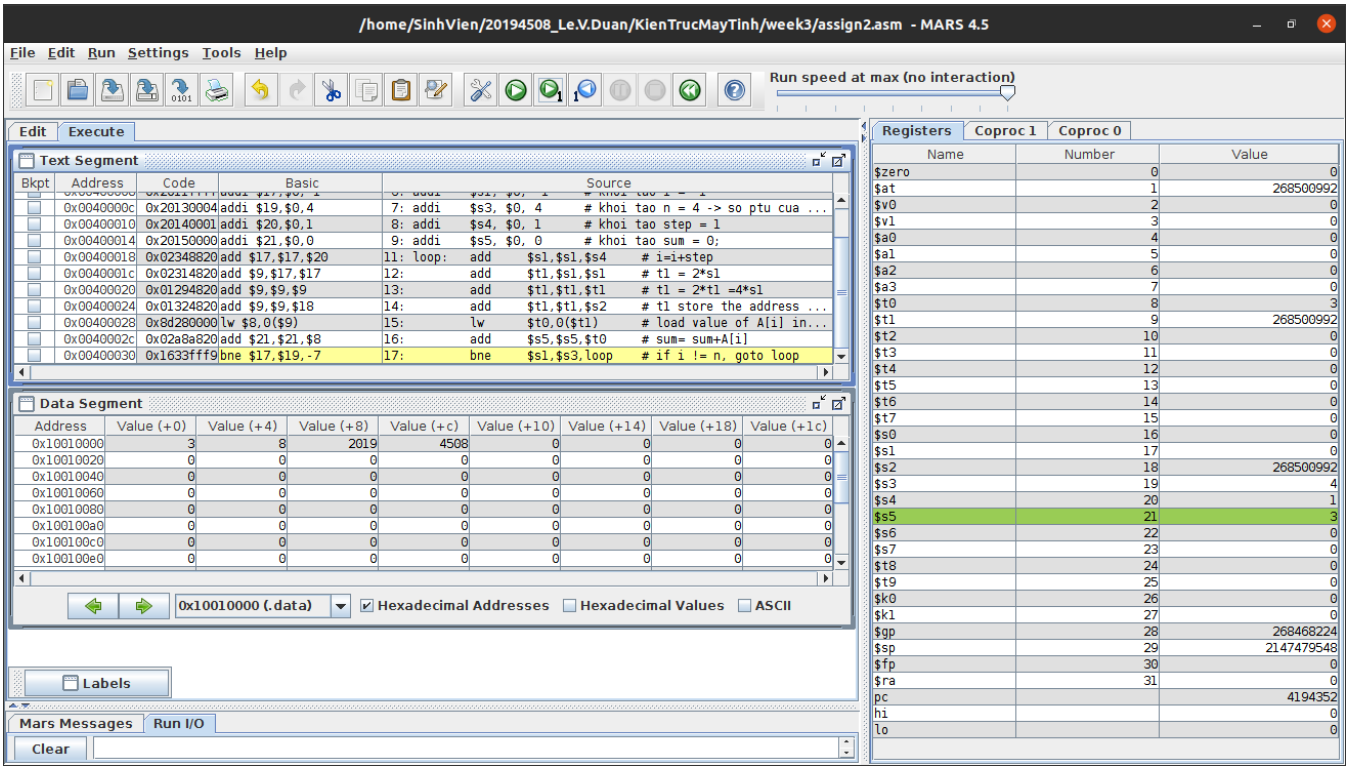
2. Assignment 2

- Mã nguồn:

```
#Laboratory 3, Home Assigment 2

.data

A:      .word  3, 8, 2019, 4508

.text

la      $s2, A

addi    $s1, $0, -1     # khoi tao i = -1

addi    $s3, $0, 4      # khoi tao n = 4 -> so ptu cua mang A
```

```
addi   $s4, $0, 1     # khoi tao step = 1
addi   $s5, $0, 0     # khoi tao sum = 0;

loop:  add    $s1,$s1,$s4  # i=i+step
add    $t1,$s1,$s1  # t1 = 2*s1
add    $t1,$t1,$t1  # t1 = 2*t1 =4*s1
add    $t1,$t1,$s2  # t1 store the address of A[i]
lw     $t0,0($t1)     # load value of A[i] in $t0
add    $s5,$s5,$t0  # sum= sum+A[i]
bne    $s1,$s3,loop# if i != n, goto loop
```

- Kết quả chạy mô phỏng sau 1 vòng lặp:



- Giải thích vòng lặp đầu tiên:

+ lệnh đầu tiên la $s2, A -> lưu địa chỉ mảng A hay chính là A[0] vào thanh ghi $s2
-> $s2 = 268500992

+ 4 lệnh tiếp theo lần lượt khai báo cách biến với giá trị khởi tạo tương ứng. Với i = -1 vì mảng bắt đầu từ chỉ số 0, step = 1 đến lặp đến phần tử ngay sau đó.

*Vòng lặp loop:

+ lệnh add $s1,$s1,$s4 -> để dịch chuyển chỉ số của mảng theo step -> $s1 = -1 + 1 = 0 -> $s1 là chỉ số mảng

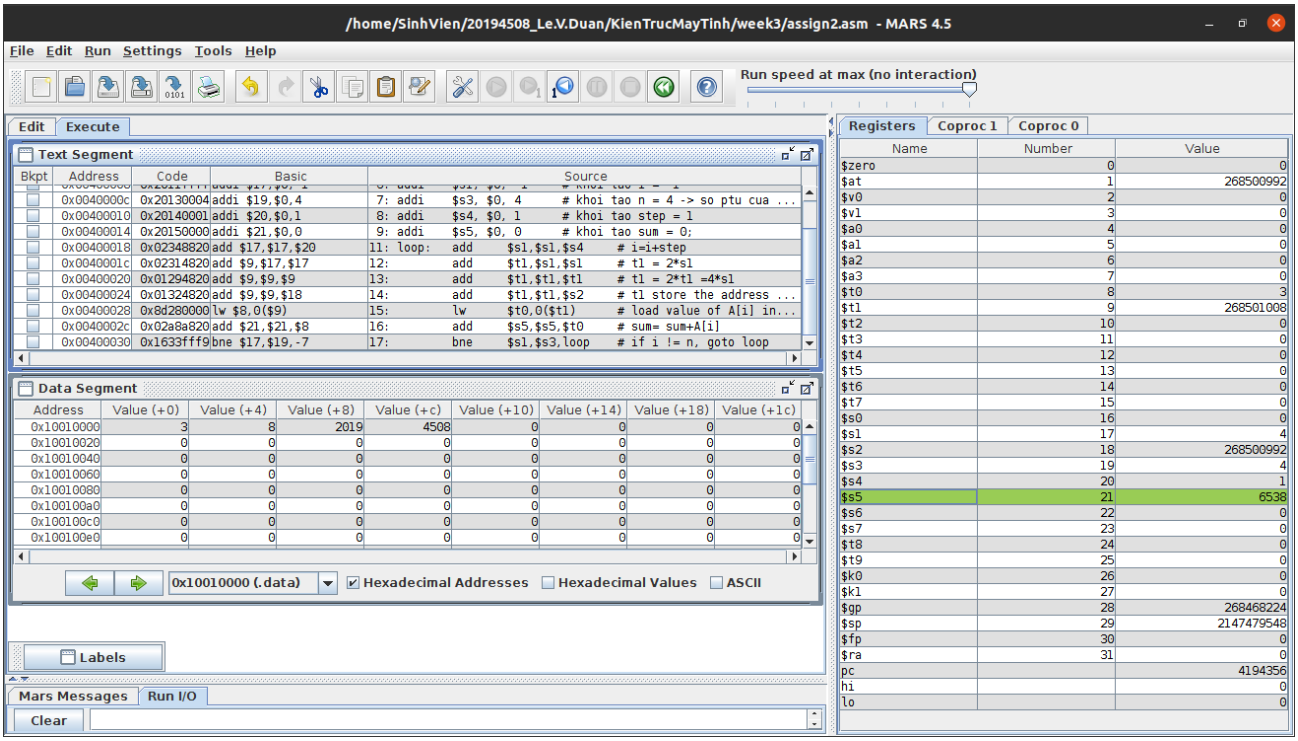
+ 2 lệnh tiếp theo với mục đích gán cho $t1 = 4*$s1 -> $t1 lưu độ dịch địa chỉ của phần tử của mảng mà các phần tử mảng có địa chỉ cách nhau 4 đơn vị -> $t1= 4 *$s1 = 0

+ lệnh lw $t0,0($t1) -> 0($t1) dùng để lưu giá trị của phần tử thứ $s1 và lệch $t1 đơn vị so với phần tử đầu tiên của mảng và lưu giá trị đó vào thanh ghi $t0 -> $t0 = 3 (= A[0])

+ lệnh add $s5,$s5,$t0 -> dùng để cộng dần giá trị các phần tử của mảng bằng cách sau mỗi lần lặp thì cộng thêm giá trị của phần tử đang xét được lưu trong thanh ghi $t0 -> $s5 = 0 + 3 = 3

+ bne $s1,$s3,loop -> là điều kiện vòng lặp so sánh với n để kết thúc vòng lặp nếu i < n thì tiếp tục đi đến nhãn loop chính là vòng lặp này.

- Sau n vòng lặp thì các giá trị của thanh ghi thay đổi theo đúng mô tả của vòng lặp đầu tiên và kết quả cho ra trong ví dụ đúng với tính toán



Sum = 3 + 8 + 2019 + 4508 = 6538 -> giá trị thanh ghi $s5

- Khi thay đổi giá trị của biến để thử các nhánh còn lại thì vẫn chạy đúng với lý thuyết đưa ra.

3. Assignment 3

-Mã nguồn:

#Laboratory Exercise 3, Home Assignment 3

.data

test: .word 1

.text

la $s0, test #load the address of test variable

lw $s1, 0($s0) #load the value of test to register $s1

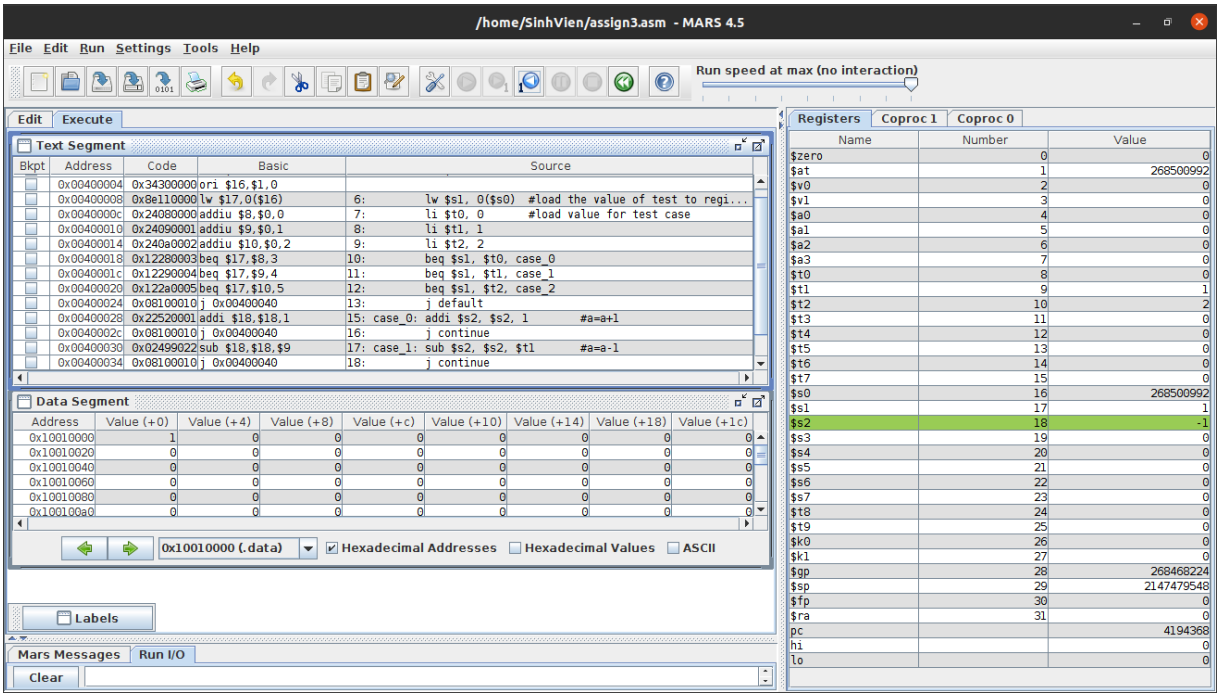
li $t0, 0 #load value for test case

li $t1, 1

li $t2, 2

```
beq $s1, $t0, case_0
beq $s1, $t1, case_1
beq $s1, $t2, case_2
j default

case_0:     addi $s2, $s2, 1     #a=a+1
j continue
case_1:     sub $s2, $s2, $t1    #a=a-1
j continue
case_2:     add $s3, $s3, $s3    #b=2*b
j continue
default:
continue:
```

- Kết quả chạy mô phỏng cho trường hợp trên:



- Giải thích:

+ 2 lệnh đầu tiên la, lw để lưu địa chỉ và giá trị của biến test vào các thanh ghi tương ứng -> $s0 = 268500992, $s1 = 1

+ 3 lệnh tiếp theo để gán các giá trị test case vào các thanh ghi -> $t0 = 0, $t1 = 1, $t2 = 2;

+ 3 lệnh beq tiếp theo là rẽ nhánh theo giá trị của test được lưu trong $s1 nếu thỏa mãn bằng test case ré nhảy tới nhãn lệnh tương ứng, nếu không có giá trị test case nào thì sẽ thực thi lệnh j default để đến nhãn lệnh default -> trong ví dụ này $s1 =1 -> nhảy đến nhãn case_1

+ tại nhãn case_1 thực hiện lệnh sub $s2, $s2, $t1 -> $s2 ban đầu có giá trị 0 và sau khi thực thi lệnh thì $s2 = 0 - 1 = -1 -> đúng như hiển thị giá trị thanh ghi $s2

+ lệnh tiếp theo thực thi j countine -> nhảy đến nhãn lệnh coutine vì không có lênh nào nên giá trị các thanh ghi không thay đổi nữa

- Tương tự các trường hợp ở case_0 ,case_2 với các giá trị biến test thỏa mãn thì giá trị các thanh ghi đều thay đổi đúng như cách giải thích ở trên.

4. Assignment 4

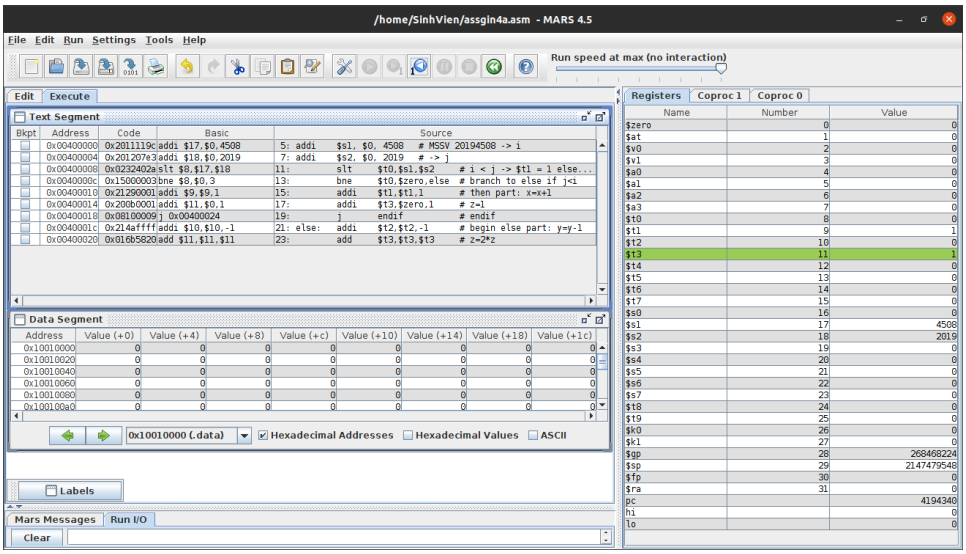
a. i < j

- Mã nguồn:

```
#Laboratory Exercise 3, Home Assignment 4 a
.text
addi   $s1, $0, 4508        # MSSV 20194508 -> i
addi   $s2, $0, 2019        # -> j
start:
slt     $t0,$s1,$s2  # i < j -> $t0 = 1 else $t0 = 0
bne    $t0,$zero,else       # branch to else if j<i
addi   $t1,$t1,1     # then part: x=x+1
```

```
addi   $t3,$zero,1  # z=1
j      endif        # endif
else:  addi   $t2,$t2,-1    # begin else part: y=y-1
add    $t3,$t3,$t3  # z=2*z
endif:
```

- Kết quả chạy:



- Khác với assign 1, khi đổi i < j thì thanh ghi $t0 = 0 do không thỏa mãn điều kiện -> không thực hiện lệnh ở nhãn else mà thực thi tiếp các lệnh phía sau

-> $t1 = 1, $t3 = 1

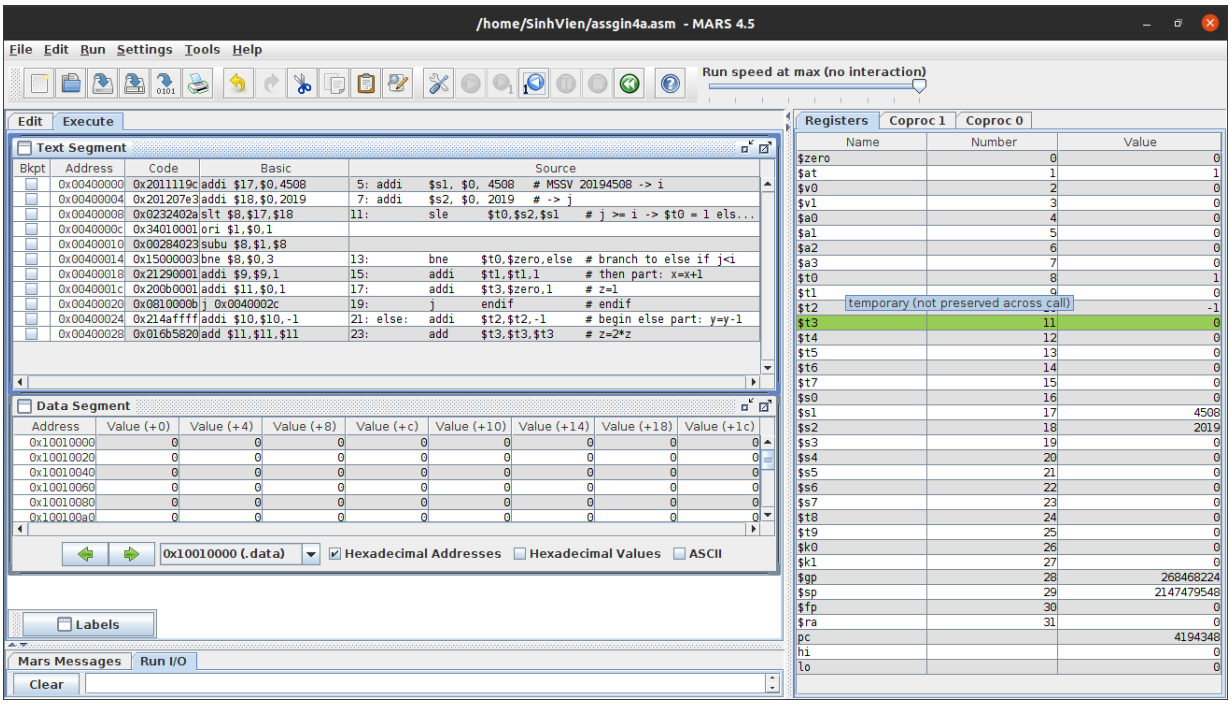
b. i >= j

- Mã nguồn:

```
#Laboratory Exercise 3, Home Assignment 4 b
.text
addi   $s1, $0, 4508        # MSSV 20194508 -> i
addi   $s2, $0, 2019        # -> j
start:
sle    $t0,$s2,$s1  # j <= i -> $t0 = 1 else $t0 = 0
```

```
bne    $t0,$zero,else       # branch to else if j<i
addi   $t1,$t1,1      # then part: x=x+1
addi   $t3,$zero,1  # z=1
j          endif          # endif
else:  addi   $t2,$t2,-1     # begin else part: y=y-1
add    $t3,$t3,$t3  # z=2*z
endif:
```

- Kết quả chạy:



- Trong ví dụ này vì j < i -> sau khi thực thi sle $t0, $s2,$s1 thì $t0 = 0 và sau đó khi thực thi lệnh bne sẽ rẽ sang nhánh else và giống như assignment 1.

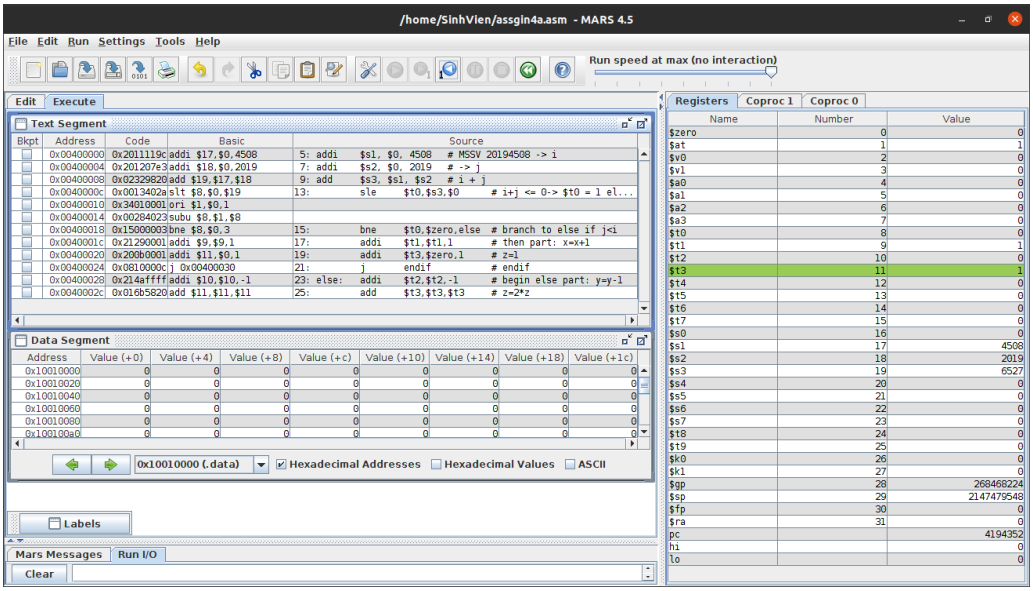
c. i + j <= 0

- Mã nguồn:

```
#Laboratory Exercise 3, Home Assignment 4 c
.text
```

```
addi   $s1, $0, 4508        # MSSV 20194508 -> i
addi   $s2, $0, 2019        # -> j
add    $s3, $s1, $s2        # i + j
start:
sle    $t0,$s3,$0   # i+j <= 0-> $t0 = 1 else $t0 = 0
bne    $t0,$zero,else       # branch to else if j<i
addi   $t1,$t1,1    # then part: x=x+1
addi   $t3,$zero,1  # z=1
j      endif        # endif
else:  addi   $t2,$t2,-1    # begin else part: y=y-1
add    $t3,$t3,$t3  # z=2*z
endif:
```

- Kết quả chạy:



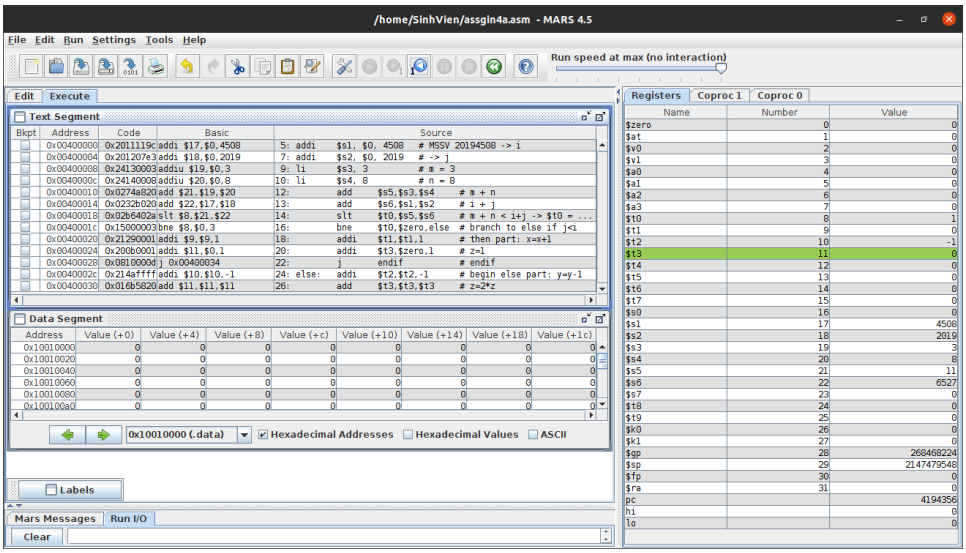
- Trong ví dụ này ta dùng thêm thanh ghi $s3 để lưu giá trị của i + j, sau đó tại câu lệnh sle để so sánh với 0 -> $t0 = 0 do i + j > 0 -> thực thi lệnh ngay sau đó và không đến nhãn else

d. i + j > m + n

- Mã nguồn:

```
#Laboratory Exercise 3, Home Assignment 4 d
.text
addi   $s1, $0, 4508        # MSSV 20194508 -> i
addi   $s2, $0, 2019        # -> j
li     $s3, 3          # m = 3
li     $s4, 8          # n = 8
start:
add    $s5,$s3,$s4  # m + n
add    $s6,$s1,$s2  # i + j
slt    $t0,$s5,$s6  # m + n < i+j -> $t0 = 1 else $t0 = 0
bne    $t0,$zero,else       # branch to else if j<i
addi   $t1,$t1,1      # then part: x=x+1
addi   $t3,$zero,1  # z=1
j      endif          # endif
else:  addi   $t2,$t2,-1    # begin else part: y=y-1
add    $t3,$t3,$t3  # z=2*z
endif:
```

- Kết quả chạy:



- Dùng $s3, $s4, $s5, $s6 để lưu giá trị của m, n, m + n, i + j

- sau lệnh slt $t0, $s5, $s6 vì i + j > m + n -> $t0 = 1 -> thực thi lệnh ở nhãn else

5. Assignment 5

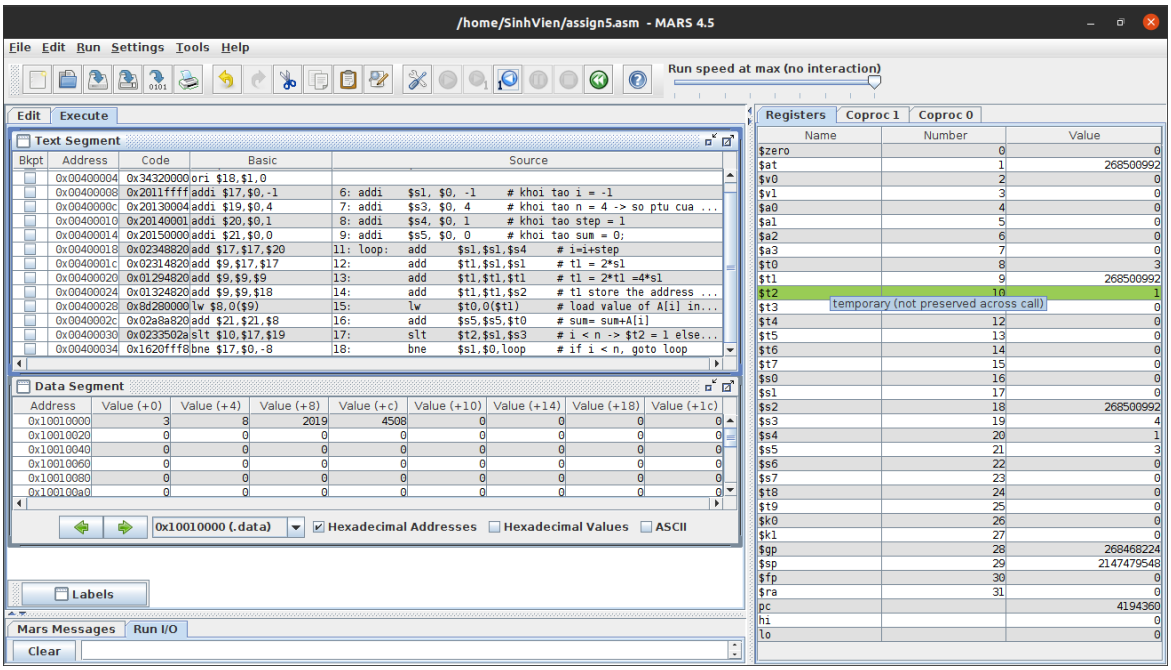
a. i < n

- Mã nguồn:

```
#Laboratory 3, Home Assigment 5 a
.data
A:      .word  3, 8, 2019, 4508
.text
la      $s2, A
addi    $s1, $0, -1     # khoi tao i = -1
addi    $s3, $0, 4      # khoi tao n = 4 -> so ptu cua mang A
addi    $s4, $0, 1      # khoi tao step = 1
addi    $s5, $0, 0      # khoi tao sum = 0;
loop:  add    $s1,$s1,$s4  # i=i+step
add     $t1,$s1,$s1  # t1 = 2*s1
add     $t1,$t1,$t1  # t1 = 2*t1 =4*s1
```

```
add    $t1,$t1,$s2  # t1 store the address of A[i]
lw     $t0,0($t1)   # load value of A[i] in $t0
add    $s5,$s5,$t0  # sum= sum+A[i]
slt    $t2,$s1,$s3  # i < n -> $t2 = 1 else $t2 = 0
bne    $s1,$0,loop  # if i < n, goto loop
```

- Kết quả chạy:



- Dùng thêm $t2 để lưu kết quả của so sánh i và n -> $t2 = 1 vì i = 0 < n = 4 -> vòng lặp khi thực hiện điều kiện bị dừng ngay lần đầu tiên -> Sum : $s5 = 3 (= A[0])

b. i <= n

c. sum >= 0

d. A[i] == 0

6. Assignment 6

- Mã nguồn:

```
1  #Assignment 6
2  #MSSV 20194713
3
4  .data
5  arr: .word 1, 12, 0, -3, -20, 19, -47, 13, 20, 15
6
7  .text
8  add $t0, $zero, $zero     #counter i
9  addi $t0, $zero, 0        #init counter
10 lw  $t3, arr($t0)         #init max variable
11 abs $s3, $t3              #absolute max variable
12
13 loop:    addi $t0, $t0, 4 #run counter
14         start:
15           lw $t4,  arr($t0)
16           abs $s4, $t4
17           slt $t5, $s4, $s3
18           bne $t5, $zero, el  #start if statement
19           lw $t3,  arr($t0)
20           abs $s3, $t3
21           j endif            #skip else statement
22         el:
23
24         endif:
25         bne $t0, 36, loop  #end loop
26
```

- Kết quả chạy: