**Bài 3. Các lệnh nhảy và lệnh rẽ nhánh**

**Họ và tên:** Nguyễn Thành Duy

**MSSV:** 20235696

**Assignment 1**

Kết quả sau khi chạy chương trình: **A screenshot of a computer

AI-generated content may be incorrect.**Sau khi chạy câu lệnh thứ nhất:



+) Thanh ghi: giá trị s1 được gán bằng 1



+) Thanh pc: 0x00400000 -> 0x00400004



Câu lệnh thứ hai tương tự câu lệnh thứ nhất: giá trị s2 được gán bằng 2, thanh ghi pc 0x00400004 -> 0x00400008





Câu lệnh thứ ba: So sánh giá trị s2 và s1, nếu s2 < s1 thì sẽ nhảy đến else còn ngược lại sẽ thực hiện câu lệnh ở ngay dưới:



Trong trường hợp này: s1 = 1, s2 = 2 nên ta thấy s2 > s1 nên sẽ thực hiện câu lệnh ở ngay phía dưới tức là then

+) Thanh ghi: 0x00400008 -> 0x0040000c



A green text on a white background

AI-generated content may be incorrect.

+) Câu lệnh sau đây thực hiện tăng giá trị của x lên 1

****

Thanh ghi:





+) Câu lệnh sau đây thực hiện gán giá trị của t3 bằng 1: Do trong riscv không có câu lệnh gán trực tiếp nên ta phải thực hiện gán thông qua phép cộng với 0 hay zero trong thanh ghi



Thanh ghi:





+) Do đã thực hiện xong các hành động ở **then** máy tính sẽ tự động chạy các câu lệnh ở ngay sau nó theo tuần tự nên để không chạy câu lệnh ngay sau là else (ở trường hợp s2 < s1) thì ta cần một lệnh nhảy tự do:



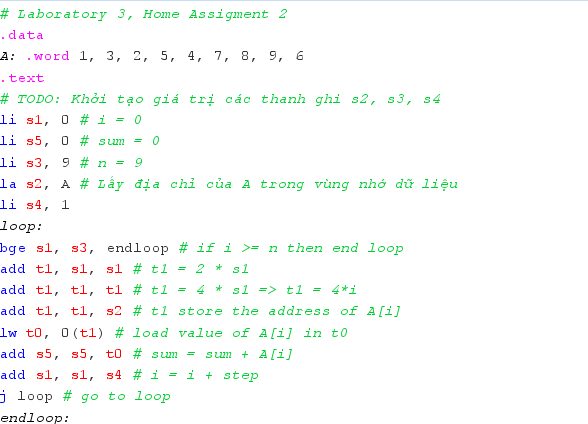
Thanh ghi:



\* Thanh ghi sau khi chạy xong chương trình:



**Assignment 2**

****

**Mục đích:** để tính tổng các phần tử của mảng A

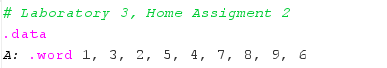
Kết quả mong muốn: sum = 1 + 3 + 2 + 5 + 4 + 7 + 8 + 9 + 6 = 45

Kết quả sau khi chạy đoạn chương trình:

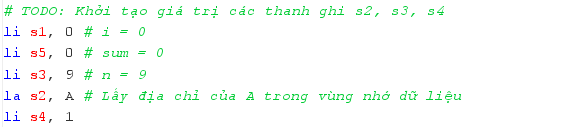
**A screenshot of a computer

AI-generated content may be incorrect.**

+) Câu lệnh đầu tiên: Khởi tạo mảng A có 9 phần tử và lần lượt có các giá trị như trong code

****

+) Ở đoạn lệnh sau:



* Câu lệnh li (load immediate) là một giả lệnh dùng để nạp một số nguyên tức thời (immediate) vào một thanh ghi

+ Câu lệnh đầu tiên: Câu lệnh sẽ gán giá trị của s1 với giá trị là 0



Thanh ghi:





+ Câu lệnh thứ hai và thứ ba và thứ tư tương tự như câu lệnh đầu tiên lần lượt gán giá trị của s3, s4, s5 là 9 (số phần tử của mảng A ban đầu khởi tạo), 1 (step trong vòng lặp ở dưới), 0 (khởi tạo tổng của mảng bằng 0)

Thanh ghi:











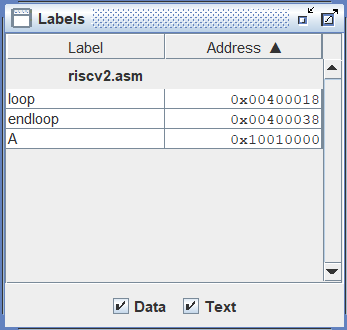


+ Ở câu lệnh thứ tư: câu lệnh **la** là lấy địa chỉ của A trong vùng nhớ dữ liệu

- Lệnh la được biên dịch thành 2 câu lệnh: auipc và addi  
+ Địa chỉ tuyệt đối được chia thành hai phần: phần cao (20-bit) và phần thấp (12-bit).  
Kết hợp hai câu lệnh này để tải một địa chỉ đầy đủ 32-bit.  
+ auipc: giúp lấy phần trên (20-bit) của địa chỉ nhãn dựa vào pc, auipc tính toán  
phần cao của địa chỉ label bằng cách lấy giá trị pc hiện tại và cộng thêm một giá trị  
offset.  
+ addi: bổ sung phần thấp (12-bit) để hoàn chỉnh địa chỉ  
-> Giúp tải địa chỉ của nhãn (label) vào thanh ghi

-> Cách này giúp chương trình có thể chạy độc lập với vị trí của bộ nhớ



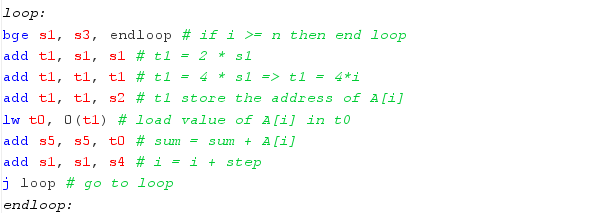


Ta thấy địa chỉ của A: 0x10010000

Thanh ghi: đã lấy được địa chỉ s2 như mong muốn





+ Ở đoạn code: 

Câu lệnh đầu tiên: điều kiện nếu s1 >= s3 thì ta sẽ kết thúc vòng lặp tức end loop. Ta có s1 = 0, s3 = 9 thấy s1 < s3 nên máy tính sẽ thực hiện câu lệnh ngay ở phía dưới

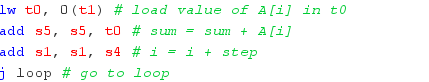


Câu lệnh: Mục đích là lấy được địa chỉ của A[i] vì mỗi phần tử cách nhau 4 bit nên ta thực hiện phép cộng **add** là vì vậy

A group of green text

AI-generated content may be incorrect.

Câu lệnh: Cộng phần tử A[i] vào sum rồi tăng step rồi lặp lại vòng lặp cho tới khi i > n sẽ thoát khỏi vòng lặp



Giá trị thanh ghi s5 khi kết thúc chương trình: s5 = 45 đúng như mong muốn



Ta thử thay đổi bộ giá trị: A = {1,3,4,5} kết quả mong muốn khi chạy sum = 1 + 3 + 4 + 5 = 13

Code: A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Giá trị thanh ghi s5 = 13 (chương trình đã chạy đúng)

**Assignment 3**

Giả lệnh:

switch(test) {

case 0:

a=a+1; break;

case 1:

a=a-1; break;

case 2:

b=2\*b; break;

}

Ta kiểm tra giá trị test nếu test = i thì chương trình nhảy tới case i: (tương ứng với test = i) và thực hiện hành động rồi thoát khỏi chương trình. Ta sẽ minh họa trên riscv:

Kết quả sau khi chạy chương trình:

**A screenshot of a computer

AI-generated content may be incorrect.**

Giá trị thanh ghi sau khi chạy câu lệnh đầu tiên: 

****

****

Giá trị thanh ghi sau khi chạy câu lệnh thứ hai:

****

****

****

Giá trị thanh ghi sau khi chạy câu lệnh thứ ba:



****

****

Giá trị thanh ghi sau khi chạy câu lệnh thứ tư:

****

****

****

Giá trị thanh ghi sau khi chạy câu lệnh thứ năm:

****

****

****

Câu lệnh tiếp theo sẽ xem giá trị test là bao nhiêu để nhảy tới câu lệnh tương ứng để thực hiện hành động



Ta thấy s1 = 0 (giá trị của test), t0 = 0 Vậy nên sẽ nhảy tới case 0 tương ứng

A group of black and green symbols

AI-generated content may be incorrect.

Do tăng giá trị của s2 lên 1 nên ở thanh ghi giá trị của s2 là 1 (đúng như mong muốn)



Ta sẽ thử một số trường hợp khác để xem chương trình chạy đúng không:

TH1: khởi tạo test = 1 thì theo mong muốn thì giá trị s2 = -1

A screenshot of a computer

AI-generated content may be incorrect.

Kết quả:

A screenshot of a computer

AI-generated content may be incorrect.

Giá trị thanh ghi s2 = -1 (như mong muốn)

**Assignment 4**

1. i = 3 , j = 4 . Nếu i < j thì t1 = 1, t3 = 1 ngược lại sẽ thực hiện câu lệnh else

Code:

|  |
| --- |
| # Laboratory Exercise 3, Home Assignment 1  .text  start:  # TODO:  li s1, 3# Khởi tạo giá trị i vào thanh ghi s1  li s2, 4# Khởi tạo giá trị j vào thanh ghi s2  # Cách 1:  blt s1, s2, then # if i < j then jump else  # Cách 2:  # slt t0, s2, s1 # set t0 = 1 if j < i else clear t0 = 0  # bne t0, zero, else # t0 != 0 means t0 = 1, jump else  then:  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: |

A screenshot of a computer

AI-generated content may be incorrect.

Kết quả: t1 = 1 , t3 = 1 (như mong muốn)

1. khởi tạo i = 4 , j = 4. Do i = j nên thực hiện câu lệnh then nên t1 = 1, t3 = 1

Code:

|  |
| --- |
| # Laboratory Exercise 3, Home Assignment 1  .text  start:  # TODO:  li s1, 4# Khởi tạo giá trị i vào thanh ghi s1  li s2, 4# Khởi tạo giá trị j vào thanh ghi s2  # Cách 1:  blt s1, s2, else # if i >= j then jump else  # Cách 2:  # slt t0, s2, s1 # set t0 = 1 if j < i else clear t0 = 0  # bne t0, zero, else # t0 != 0 means t0 = 1, jump else  then:  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: |

Kết quả:

A screenshot of a computer

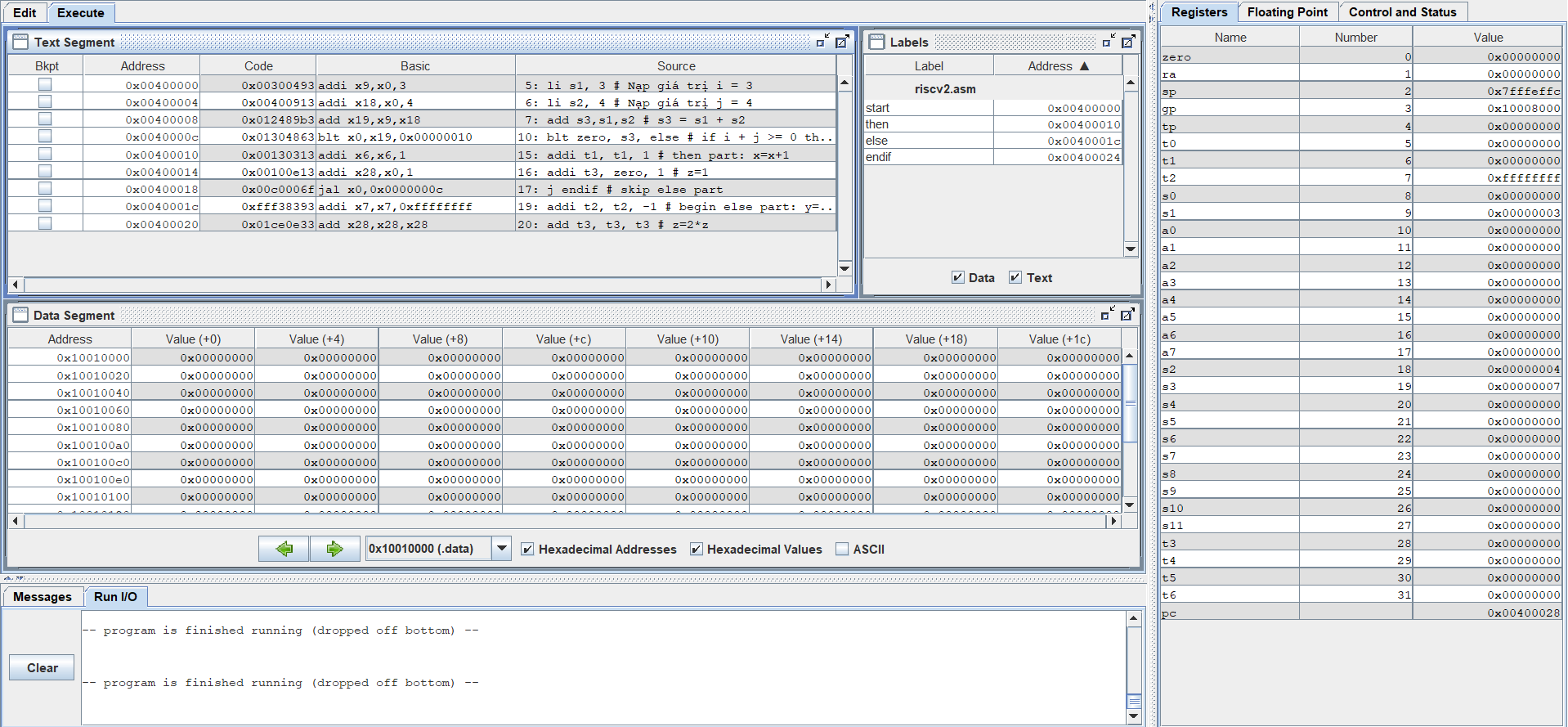
AI-generated content may be incorrect.

Kết quả: t1 = 1, t3 = 1(như mong muốn)

1. Ta sẽ sử dụng điều kiện 0 < i + j sẽ thực hiện câu lệnh else còn i + j <= 0 sẽ thực hiện then. Khởi tạo i = 3, j = 4 -> i = j > 0 nên t2 = -1, t3 = 0

A screenshot of a computer program

AI-generated content may be incorrect.



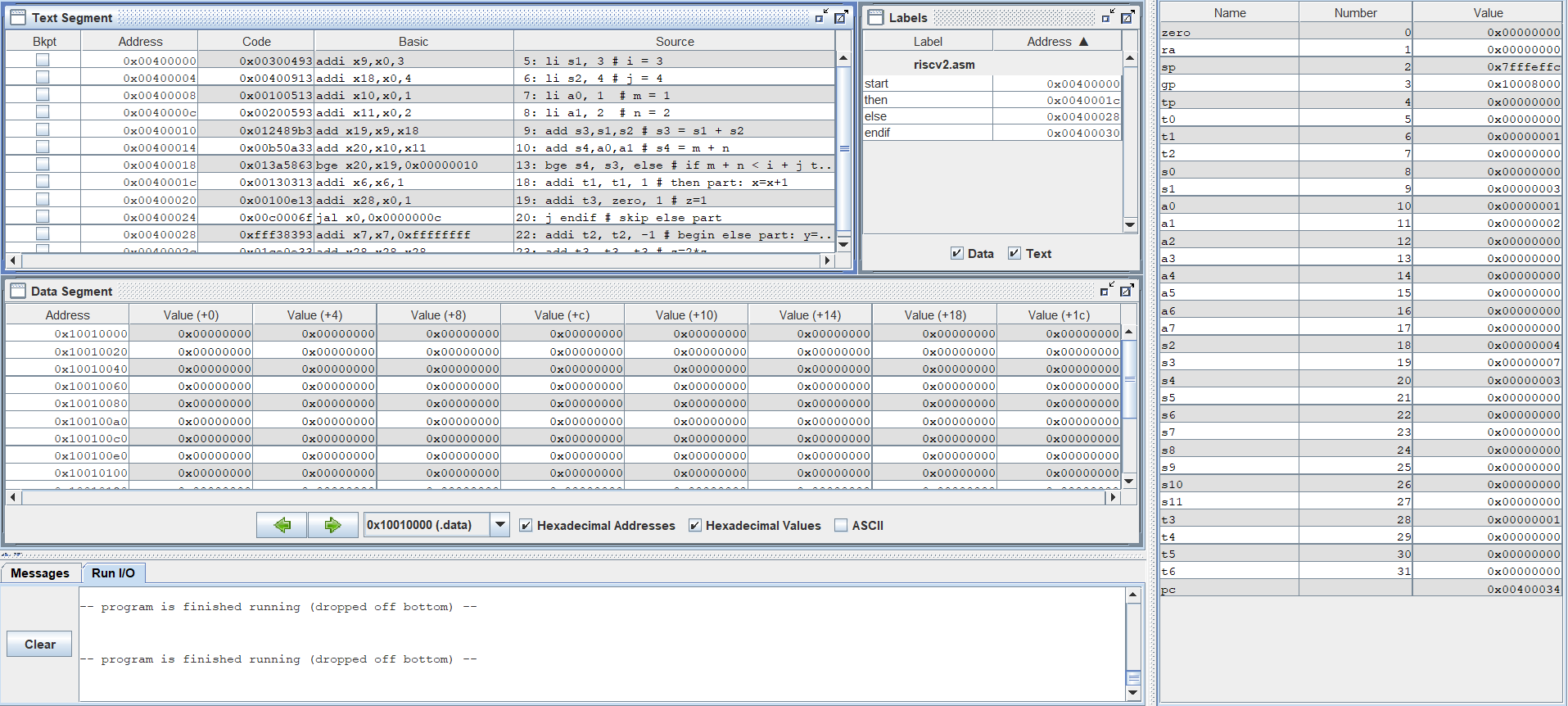
Kết quả: đúng như mong muốn

1. Nếu m + n >= i + j sẽ thực hiện else còn i + j > m + n thực hiện then. Ta có m + n = 3, i + j = 7 nên sẽ thực hiện then

A white screen with green text

AI-generated content may be incorrect.

Kết quả:



**Assignment 5**

1. i > n hay n < i thì chương trình kết thúc

|  |
| --- |
| # Laboratory 3, Home Assigment 2  .data  A: .word 1, 3, 2, 5  .text  # TODO: Khởi tạo giá trị các thanh ghi s2, s3, s4  li s1, 0 # i = 0  li s5, 0 # sum = 0  li s3, 9 # n = 4  la s2, A # Lấy địa chỉ của A trong vùng nhớ dữ liệu  li s4, 1 # s4 = 1  loop:  blt s3, s1, endloop # if i > n then end loop  add t1, s1, s1 # t1 = 2 \* s1  add t1, t1, t1 # t1 = 4 \* s1 => t1 = 4\*i  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]  add s1, s1, s4 # i = i + step  j loop # go to loop  endloop: |

**A screenshot of a computer

AI-generated content may be incorrect.**

Kết quả: tính tổng các phần tử mảng A (sum = 11)

1. sum < 0 thì chương trình kết thúc

**Chương trình:**

|  |
| --- |
| # Laboratory 3, Home Assigment 2  .data  A: .word 1, 3, -5, 6  .text  # TODO: Khởi tạo giá trị các thanh ghi s2, s3, s4  li s1, 0 # i = 0  li s5, 0 # sum = 0  li s3, 9 # n = 4  la s2, A # Lấy địa chỉ của A trong vùng nhớ dữ liệu  li s4, 1 # s4 = 1  loop:  blt s5, zero, endloop # if sum < 0 then end loop  add t1, s1, s1 # t1 = 2 \* s1  add t1, t1, t1 # t1 = 4 \* s1 => t1 = 4\*i  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]  add s1, s1, s4 # i = i + step  j loop # go to loop  endloop: |

**A screenshot of a computer

AI-generated content may be incorrect.**

Kết quả: sum = 1 + 3 – 5 = -1 < 0 nên chương trình kết thúc.

1. A[i] == 0 chương trình kết thúc

**Chương trình:**

.data

A: .word 1, 3, 0, 5, 4, 7, 8, 9, 6

.text

li s1, 0 # i = 0

la s2, A # dia chi so dau tien

li s3, 9 # n = 9

li s4, 1 # step = 1

li s5, 0 # sum = 0

addi s3, s3, 1

loop:

add t1, s1, s1

add t1, t1, t1

add t1, t1, s2

lw t0, 0(t1)

beq t0, zero, endloop

add s5, s5, t0

add s1, s1, s4

j loop

endloop:

**Assignment 6**

**Chương trình:**

.data

A: .word -10 20 -30 4 0x80000000 -90 #Mang

n: .word 6 #so phan tu

.text

la t0, A

lw t1, n

li t2, 0 # i = 0

li t3, 1 # step = 1

li t4, 0 # max ||

loop:

bge t2, t1, endloop

add s0, t2, t2

add s0, s0, s0

add s0, s0, t0

li s3, 0x80000000

lw s1, 0(s0)

bge s1, zero, continue # neu >= 0 thi tiep tuc

beq s1, s3, extra # neu gap 0x80000000 thi thoat luon

sub s1, zero, s1 # doi dau

continue:

blt t4, s1, update #neu max < s1 thi cap nhat max

add t2, t2, t3 #tang i

j loop

update:

addi t4, s1, 0 #cap nhat max bang s1

add t2, t2, t3

j loop

endloop:

li a7, 1

mv a0, t4 # Đưa max vào a0

ecall # Gọi syscall để in

j end

extra:

li a7, 1

mv a0, s1

ecall

end:

Kết luận:

* Các lệnh Branch thực hiện nhảy khi thỏa mãn một điều kiện nào đó.

Được sử dụng trong các cấu trúc if/else, các cấu trúc lặp.

* Các lệnh Jump thực hiện nhảy mà không cần điều kiện nào cả. Được sử dụng để gọi hàm.