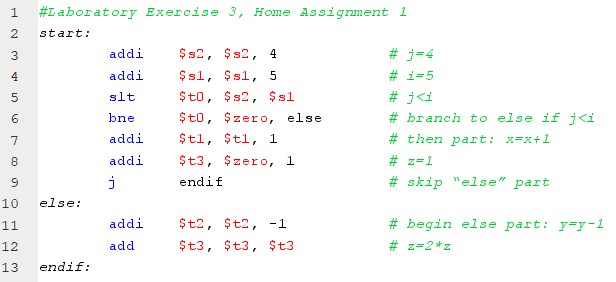
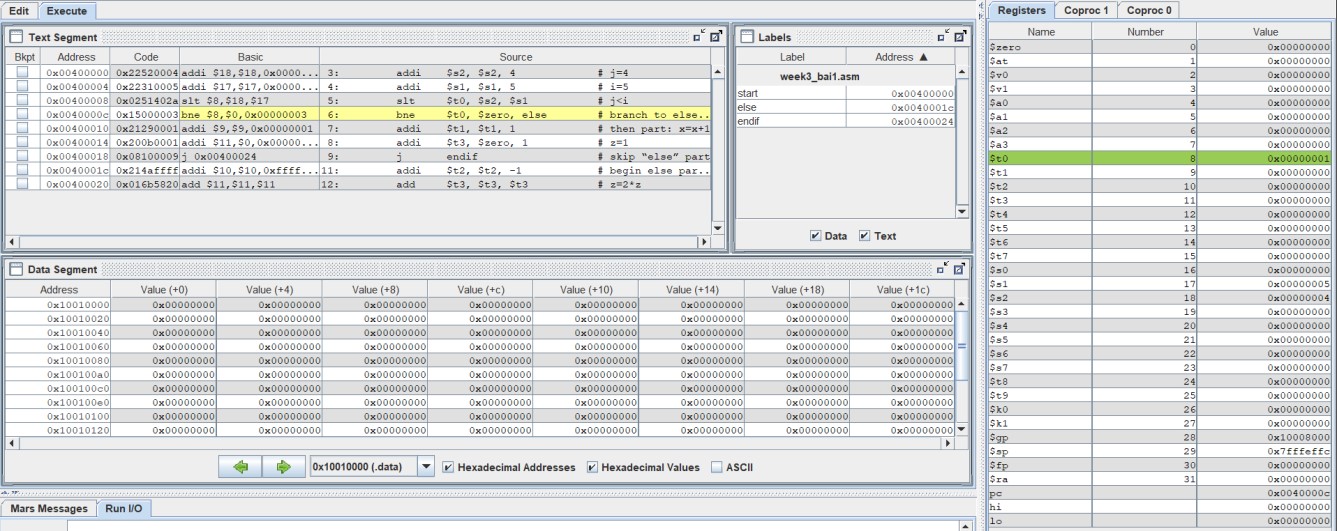
**Bài thực hành tuần 3**

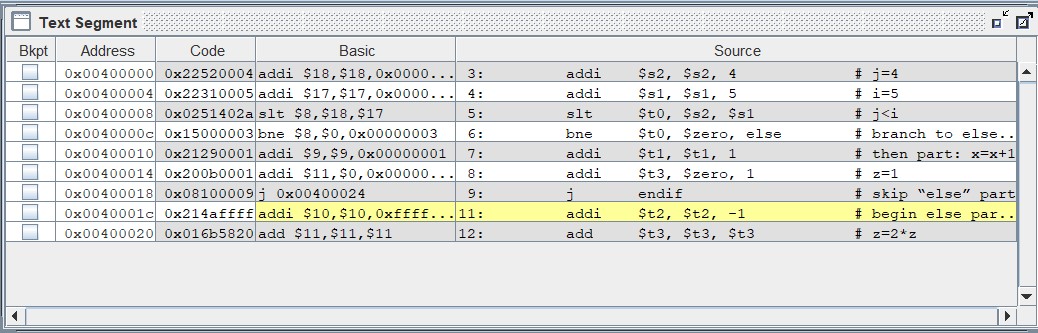
**Lớp: 130937 – Học phần: Thực hành Kiến trúc máy tính Bài 1:**



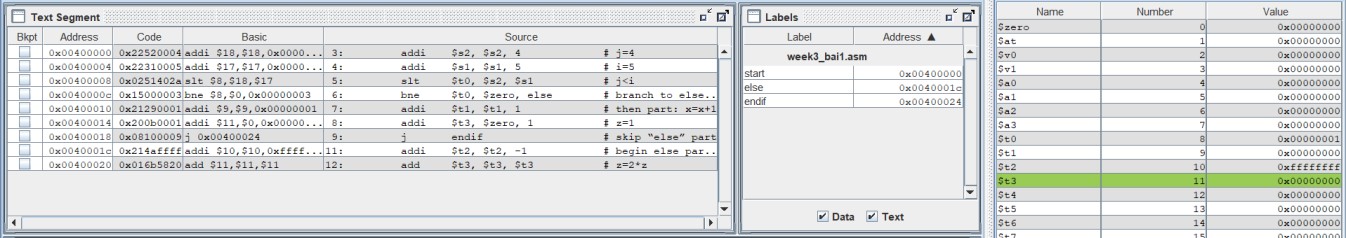
* Đầu tiên ta có j = 4 và i = 5. Sau đó đến dòng 5 lệnh slt sẽ thực hiện so sánh j và i. Ở đây ta có j < i nên $t0 = 1.



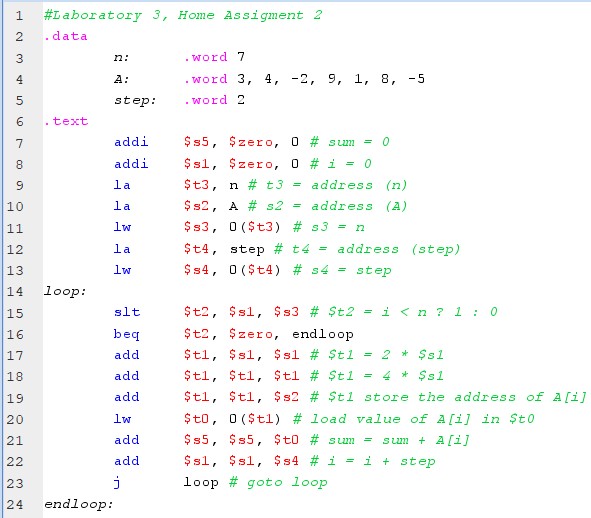
* Ở dòng 6, lệnh bne thực hiện so sánh xem $t0 có khác 0 không, tức là kiểm tra j < i, ở đây j < i nên chương trình sẽ nhảy đến *else*, tức là chương trình thực hiện tiếp ở dòng 11.



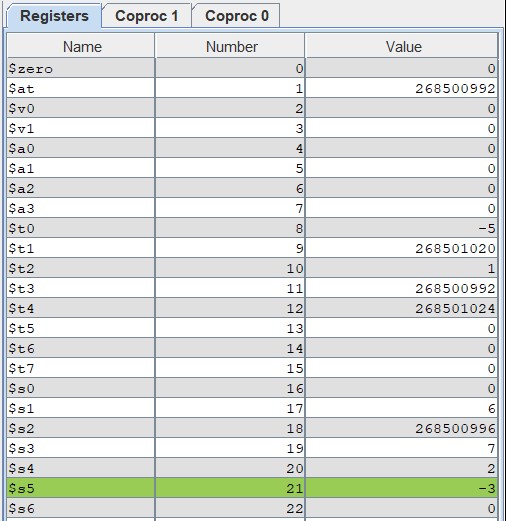
* Ở dòng 11, lệnh addi sẽ thực hiện trừ $t2 bớt đi 1, tức là trừ y bớt đi 1. Dòng 12 thì lệnh add sẽ thực hiện cộng thanh ghi $t3 với $t3 và kết quả lưu ở $t3, tức là nhân đôi giá trị ở thanh ghi $t3 hay giá trị z.



**Bài 2:**

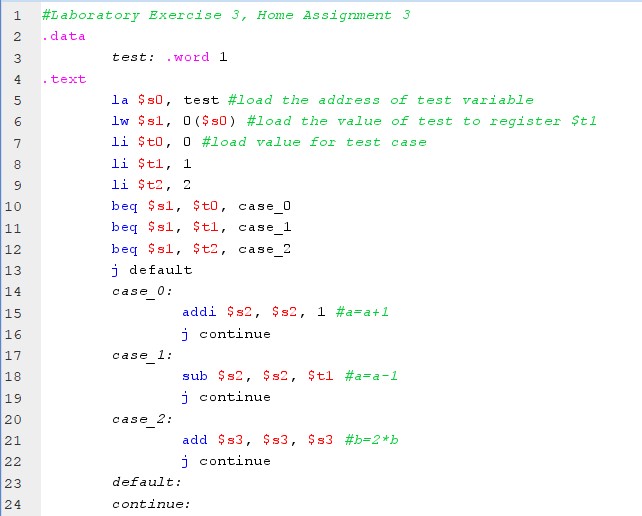


* Từ dòng 7 đến 13:
  + - $s5 (sum) được gán bằng 0.
    - $s1 (i) được gán bằng 0.
    - $t3 được gán bằng địa chỉ của n.
    - $s2 được gán bằng địa chỉ của mảng A (tương đương địa chỉ phần từ A[0]).
    - $t4 được gán bằng địa chỉ của step.
    - Lệnh lw giúp cho các thanh ghi $s3, $s4 được nạp giá trị của n và step.
* Phần loop:
  + - Lệnh slt dùng để so sánh $s1 và $s3 (tức là i và n). Khi i vẫn nhỏ hơn n thì $t2 = 1.
    - Lệnh beq dùng để so sánh $t2 và 0, nếu $t2 = 0 (i ≥ n) thì chương trình nhảy xuống endloop (thoát vòng lặp). Còn khi i < n thì $t2 = 1, thực hiện tuần tự tiếp.
    - 3 lệnh add sau đó, thanh ghi của $t1 sẽ được gán bằng $s2 + 4 \* $s1, tức là [địa chỉ mảng A] + 4 \* i. Như vậy thanh ghi $t1 sẽ chứa địa chỉ của phần tử A[i].
    - Lệnh lw giúp cho ngăn nhớ có địa chỉ là $t1 + 0 (tức là điạ chỉ của A[i]) được nạp vào $t0. Khi đó $t0 = A[i].
    - Lệnh add dòng 21: $s5 = $s5 + $t0, nghĩa là giá trị sum được cộng với A[i].
    - Lệnh add dòng 22: $s1 = $s1 + $s4, nghĩa là giá trị i được cộng với step. - Lệnh j dòng 23: quay lại phần loop, có nghĩa là tiếp tục vòng lặp.



* Với các giá trị n, step, các phần tử mảng A như trên, thì sum ($s5) có giá trị là 3 + (−2) + 1 + (−5) = −3.

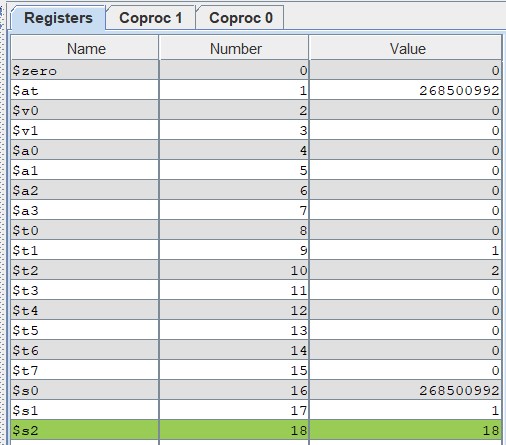
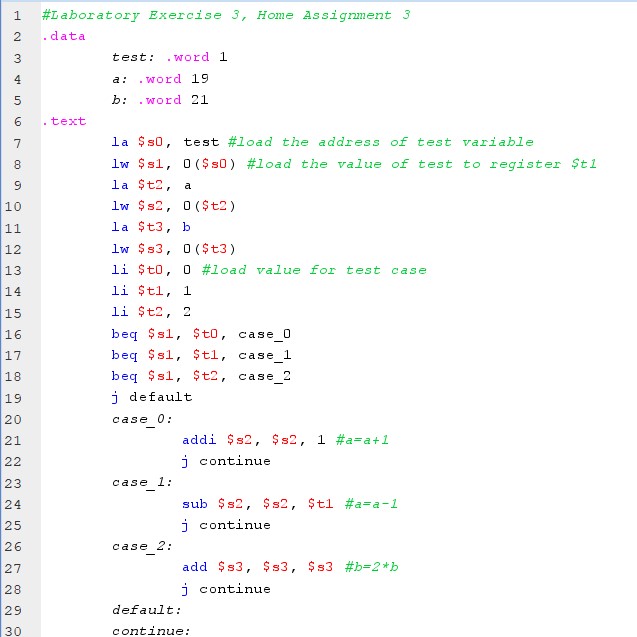
**Bài 3:**



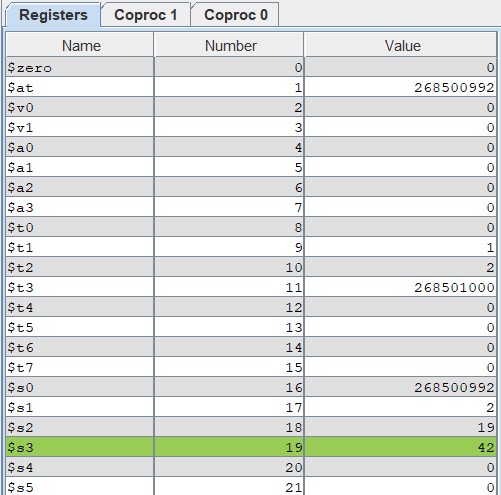
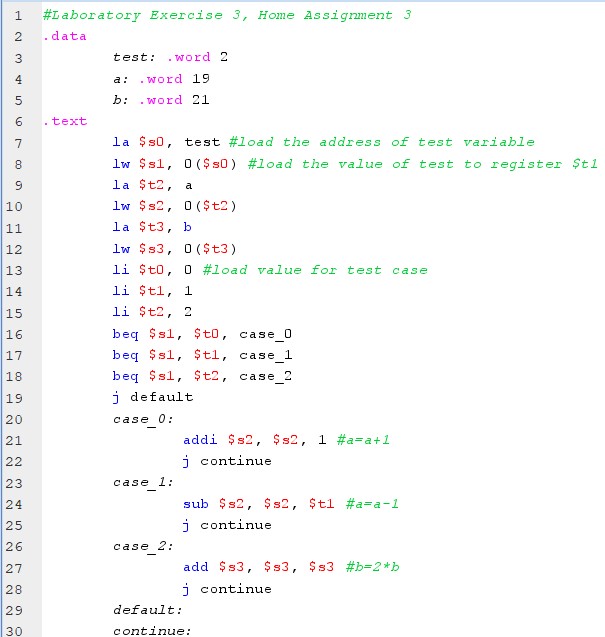
* Dòng 5 và 6: $s0 sẽ nạp giá trị là địa chỉ của test, sau đó $s1 sẽ nạp giá trị tại ngăn nhớ có địa chỉ là $s0, tức là $s1 = test.
* Dòng 7, 8, 9: $t0, $t1, $t2 nạp giá trị 0, 1, 2.
* Dòng 10 – 13:
  + Dòng 10 – 12: So sánh $s1 và $t0, $t1, $t2, tức là so sánh giá trị test với

0, 1, 2. Ứng với mỗi trường hợp thì chương trình sẽ nhảy đến lệnh được gắn nhãn tương ứng.

* + Dòng 13: Khi giá trị test không phải là 0, 1, 2 thì sẽ không thực hiện case nào cả mà nhảy luôn đến default.
* Ứng với mỗi phần lệnh được gắn nhãn case\_0, case\_1, case\_2, khi thực hiện xong sẽ có lệnh jump để chương trình nhảy đến phần continue để tiếp tục chương trình.



* Trong ví dụ này: test = 1, $s2 = a = 19, khi đó chương trình sẽ nhảy đến case\_1 thực hiện lệnh ở dòng 18: $s2 = $s2 – 1, khi đó $s2 = 19 – 1 = 18.



* Trong ví dụ này: test = 1, $s3 = b = 21, khi đó chương trình sẽ nhảy đến case\_1 thực hiện lệnh ở dòng 18: $s3 = $s3 + $s3, khi đó $s3 = 21 + 21 = 42.

**Bài 4:**

a)

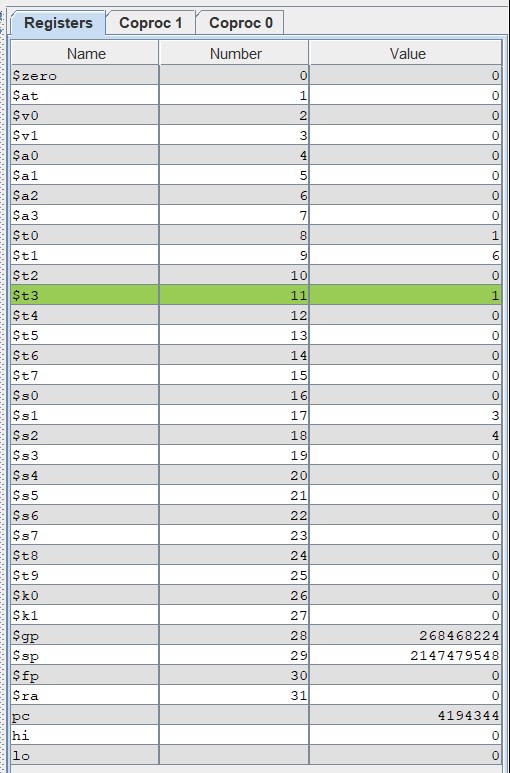
* Code:

#Laboratory Exercise 3, Home Assignment 4a start:

|  |  |
| --- | --- |
| addi $t1, $t1, 5 | # x=5 |
| addi $s2, $s2, 4 | # j=4 |
| addi $s1, $s1, 3 | # i=3 |
| 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  else: | # skip “else” part |
| addi $t2, $t2, -1 | # begin else part: y=y-1 |
| add $t3, $t3, $t3 | # z=2\*z |

endif:

* Kết quả:



b)

* Code:

#Laboratory Exercise 3, Home Assignment 4b start:

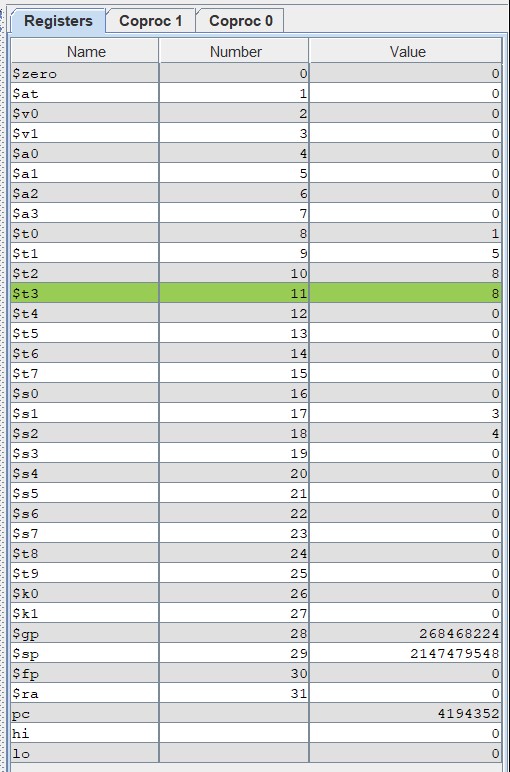
|  |  |
| --- | --- |
| addi $t1, $t1, 5 | # x=5 |
| addi $t2, $t2, 9 | # y=9 |
| addi $t3, $t3, 4 | # z=4 |
| addi $s2, $s2, 4 | # j=4 |
| addi $s1, $s1, 3 | # i=3 |
| slt $t0, $s1, $s2 | # i<j |
| bne $t0, $zero, else | # branch to else if i<j |
| addi $t1, $t1, 1 | # then part: x=x+1 |
| addi $t3, $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:

* Kết quả:



c)

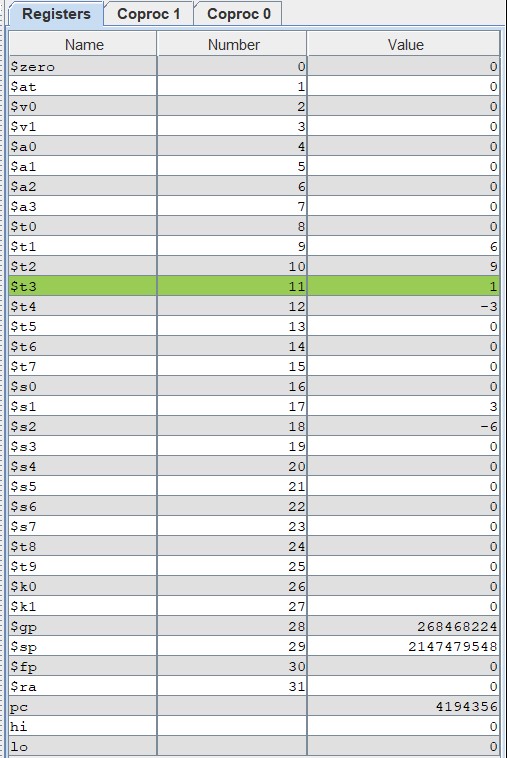
* Code:

#Laboratory Exercise 3, Home Assignment 4c start:

|  |  |  |
| --- | --- | --- |
| addi $t1, $t1, 5 |  | # x=5 |
| addi $t2, $t2, 9 |  | # y=9 |
| addi $t3, $t3, 4 |  | # z=4 |
| addi $s2, $s2, -6 |  | # j=-6 |
| addi $s1, $s1, 3 |  | # i=3 |
| add $t4, $s1, $s2 |  | # $t4=i+j |
| slt $t0, $0, $t4 | | # i+j>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  else: | | # skip “else” part | |
| addi $t2, $t2, -1 | | # begin else part: y=y-1 | |
| add $t3, $t3, $t3 | | # z=2\*z | |

endif:

* Kết quả:



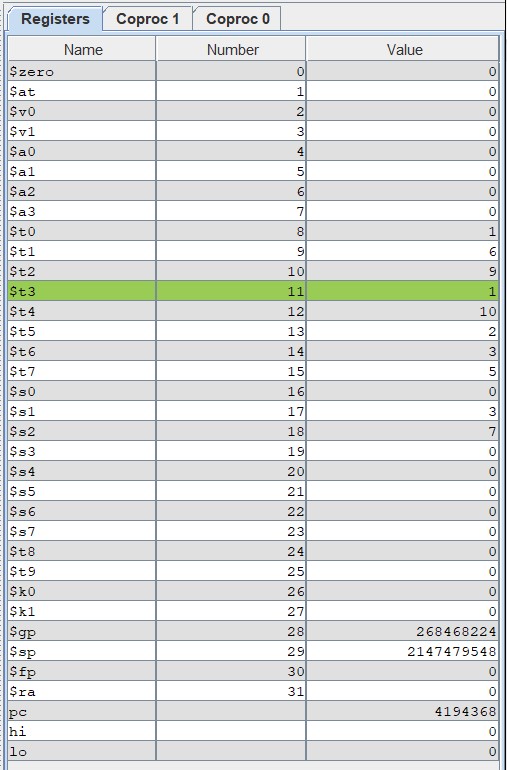
d)

* Code:

#Laboratory Exercise 3, Home Assignment 4d start:

|  |  |  |
| --- | --- | --- |
| addi $t1, $t1, 5 | # x=5 | |
| addi $t2, $t2, 9 | # y=9 | |
| addi $t3, $t3, 4 | | | # z=4 | |
| addi $s2, $s2, 7 | | | # j=7 | |
| addi $s1, $s1, 3 | | | # i=3 | |
| addi $t5, $t5, 2 | | | # m=2 | |
| addi $t6, $t6, 3 | | | # n=3 | |
| add $t4, $s1, $s2 | | | # $t4=i+j | |
| add $t7, $t5, $t6 | | | # $t7=m+n | |
| slt $t0, $t7, $t4 | | | # i+j>m+n | |
| beq $t0, $zero, else | | | # branch to else if i+j>m+n | |
| addi $t1, $t1, 1 | | | # then part: x=x+1 | |
| addi $t3, $zero, 1 | | | # z=1 | |
| j endif  else: | | | # skip “else” part | |
| addi $t2, $t2, -1 | | | # begin else part: y=y-1 | |
| add $t3, $t3, $t3 | | | # z=2\*z | |

endif:



**Bài 5:**

a) Đã làm ở bài 2

b)

• Code:

#Laboratory 3, Home Assigment 5b

.data

n: .word 7

A: .word 3, 4, -2, 9, 1, 8, -5, 2 step: .word 1

.text addi $s5, $zero, 0 # sum = 0 addi $s1, $zero, 0 # i = 0 la $t3, n # t3 = address (n) la $s2, A # s2 = address (A) lw $s3, 0($t3) # s3 = n la $t4, step # t4 = address (step) lw $s4, 0($t4) # s4 = step

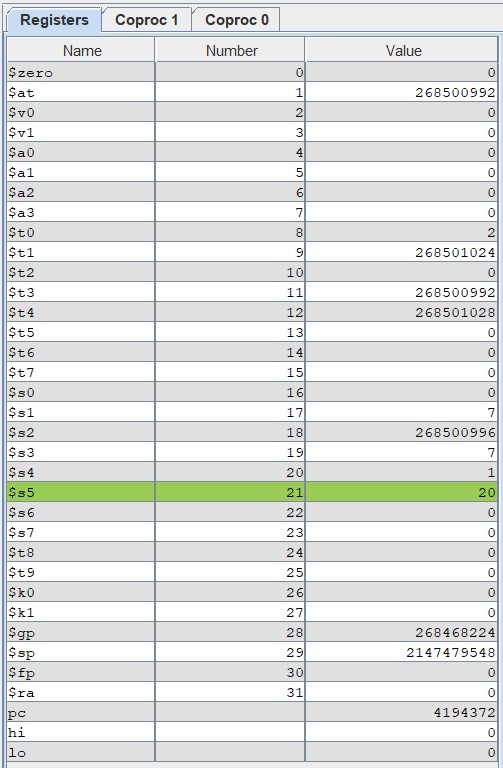
loop:

slt $t2, $s3, $s1 # $t2 = i > n ? 1 : 0 bne $t2, $zero, endloop # i > n -> 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) # load value of A[i] in $t0 add $s5, $s5, $t0 # sum = sum + A[i] add $s1, $s1, $s4 # i = i + step

j loop # goto loop

endloop:

* Kết quả:



c)

* Code:

#Laboratory 3, Home Assigment 5c

.data

n: .word 7

A: .word -3, -4, -2, 12, 1, 8, -5 step: .word 1

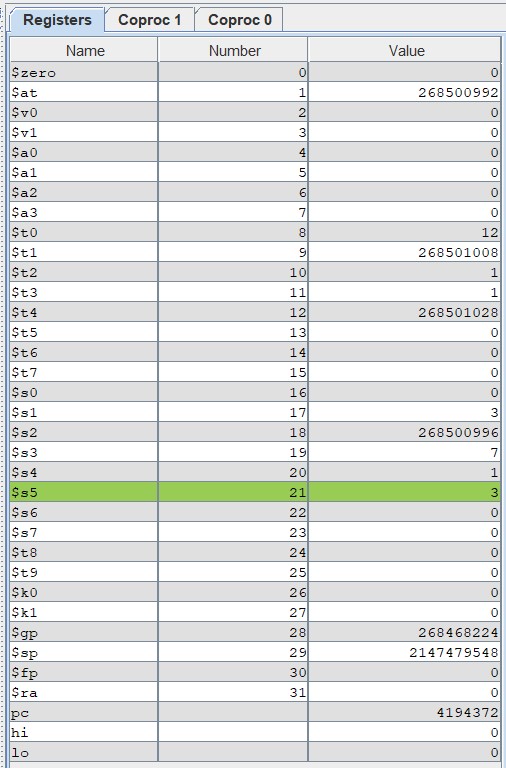
.text

|  |  |
| --- | --- |
| addi $s5, $zero, 0 | # sum = 0 |
| addi $s1, $zero, 0 | # i = 0 |
| la $t3, n | # t3 = address (n) |
| la $s2, A | # s2 = address (A) |
| lw $s3, 0($t3) | # s3 = n |
| la $t4, step | # t4 = address (step) |
| lw $s4, 0($t4) | # s4 = step |

loop:

|  |  |
| --- | --- |
| slt $t2, $s1, $s3 beq $t2, $zero, endloop | # $t2 = i < n ? 1 : 0 |
| 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) | # load value of A[i] in $t0 |
| add $s5, $s5, $t0 | # sum = sum + A[i] |
| add $s1, $s1, $s4 | # i = i + step |
| slt $t3, $s5, $0 | # sum < 0 |
| beq $t3, $0, endloop | # sum >=0 -> endloop |
| j loop | # goto loop |

endloop: • Kết quả:



d)

* Code:

#Laboratory 3, Home Assigment 5d

.data

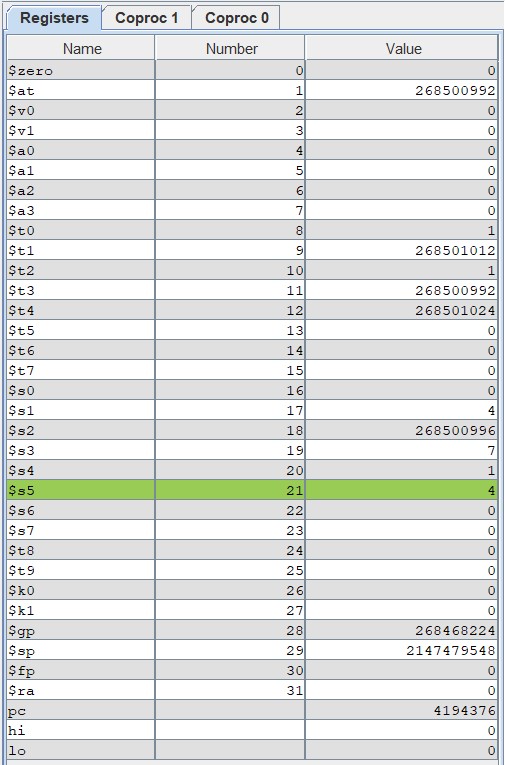
n: .word 7

A: .word -3, -4, -2, 12, 1, 0, -5 step: .word 1 .text

|  |  |
| --- | --- |
| addi $s5, $zero, 0 | # sum = 0 |
| addi $s1, $zero, 0 | # i = 0 |
| la $t3, n | # t3 = address (n) |
| la $s2, A | # s2 = address (A) |
| lw $s3, 0($t3) | # s3 = n |
| la $t4, step | # t4 = address (step) |
| lw $s4, 0($t4)  loop: | # s4 = step |
| slt $t2, $s1, $s3 beq $t2, $zero, endloop | # $t2 = i < n ? 1 : 0 |
| 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) | # load value of A[i] in $t0 |
| beq $t0, $0, endloop | # A{i]==0 -> endloop |
| add $s5, $s5, $t0 | # sum = sum + A[i] |
| add $s1, $s1, $s4 | # i = i + step |
| j loop | # goto loop |

endloop:

* Kết quả:



**Bài 6:**

* Code:

#Laboratory 3, Home Assigment 5d

.data

|  |  |
| --- | --- |
| n: .word 7  A: .word -3, 4, 2, -12, 1, 0, -5 | |
| .text |  |
| la $s0, n | # $s0 = address (n) |
| lw $s1, 0($s0) | # $s1 = n |
| la $s2, A | # $s2 = address (A) |
| lw $t2, 0($s2) | # max = A[0] |
| slt $t5, $t2, $0 | # $t2<0 |
| beq $t5, $0, loop | # $t2>=0 -> loop |
| sub $t2, $0, $t2 | # $t2<0 -> $t2 = -$t2 (max = $t2 = |A[0]|) |

loop:

|  |  |
| --- | --- |
| slt $t7, $t0, $s1 # $t7 = i < n ? 1 : 0 beq $t7, $zero, endloop | |
| add $t1, $t0, $t0 | # $t1=2\*i, $t0 = i |
| add $t1, $t1, $t1 | # $t1=4\*i |
| add $t1, $s2, $t1 | # $t1=$s2+4\*i |
| lw $s3, 0($t1) addi $t0, $t0, 1 | # $s3=A[i] |
| slt $t4, $s3, $0 | # A[i]<0 |
| beq $t4, $0, cont | # A[i]>=0 -> cont |
| sub $s3, $0, $s3 cont: | # $s3=A[i]<0 -> $s3 = -$s3 ($s3 = |A[i]|) |
| slt $t6, $t2, $s3 | # max < |A[i]| -> $t6=1 |
| beq $t6, $0, loop | # max >= |A[i]| -> loop |
| add $t2, $0, $s3 | # max < |A[i]| -> max = |A[i]| |
| j loop # jump loop | |

endloop:

* Kết quả: (max = 12)

