

BÁO CÁO THỰC HÀNH
IT3280 –THỰC HÀNH KIẾN TRÚC MÁY TÍNH
Tuần 03: LOAD/ STORE , JUMP & BRANCH INSTRUCTIONS.

Họ và tên	Đinh Ngọc Khánh Huyền
Mã số sinh viên	20225726

Assignment 1

a. $i \leq j$

* Khởi tạo với $i=1$, $j=2$ ($i \leq j$), $x=3$, $y=4$, $z=5$

- Mã nguồn:

The screenshot shows the MARS MIPS simulator interface. The assembly code in the editor is as follows:

```
1 .data
2 i: .word 1
3 j: .word 2
4 .text
5 la $s0, i
6 lw $s1, 0($s0)
7 la $s0, j
8 lw $s2, 0($s0)
9 li $t1, 3
10 li $t2, 4
11 li $t3, 5
12 start:
13 slt $t0, $s2, $s1
14 bne $t0, $zero, else
15 addi $t1, $t1, 1
16 addi $t3, $zero, 1
17 j endif
18 else:
19 addi $t2, $t2, -1
20 add $t3, $t3, $t3
21 endif:
```

The Registers window on the right shows the following values:

Name	Number	Value
\$zero	0	0
\$at	1	268500592
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	4
\$t2	10	4
\$t3	11	1
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	268500596
\$s1	17	1
\$s2	18	2
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$t8	24	0
\$t9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
pc		4194368
hi		0
lo		0

The console at the bottom shows the message: "Reset: reset completed." and "-- program is finished running (dropped off bottom) --".

- Nhận xét:

- Khởi tạo giá trị $i=1$ ứng với thanh ghi $\$s1$ và giá trị của $j=2$ ứng với thanh ghi $\$s2$
- Khởi tạo giá trị x, y, z ứng với thanh ghi $\$t1, \$t2, \$t3$
- Ở câu lệnh “`slt $t0, $s2, $s1`”, điều kiện $j < i$ không đúng nên thanh ghi $\$t0 = 0$ xuyên suốt chương trình.
- Khi đến câu lệnh `bne` (**B**ranch **I**f **N**ot **E**qual), giá trị của thanh ghi $\$t0$ bằng với giá trị của $\$zero$ nên không thực hiện rẽ nhánh đến `else`, vì vậy, chương trình sẽ tiếp tục thực hiện phần code ở `start`.
- $x = x + 1, z = 1 \Rightarrow \$t1 = 4; \$t3 = 1; \$t2$ giữ nguyên giá trị

=> Đúng với lý thuyết đưa ra

- Nhảy xuống endif và kết thúc

*Khởi tạo với $i=2, j=1$ ($i>j$), $x=3, y=4, z=5$

- Mã nguồn:

The screenshot shows the MARS MIPS simulator interface. The main window displays assembly code for a file named 'lab3_huyen'. The code includes data declarations for variables i, j, x, y, and z, followed by a loop structure with conditional jumps. The 'Registers' window on the right shows the current state of MIPS registers, with \$t0, \$t1, \$t2, and \$t3 containing values 1, 3, 10, and 10 respectively. The 'Mars Messages' window at the bottom shows a message indicating the program has finished running.

```
1 .data
2     i: .word 2
3     j: .word 1
4 .text
5     la $s0, i
6     lw $s1, 0($s0)
7     la $s0, j
8     lw $s2, 0($s0)
9     li $t1, 3
10    li $t2, 4
11    li $t3, 5
12 start:
13    slt $t0, $s2, $s1
14    bne $t0, $zero, else
15    addi $t1, $t1, 1
16    addi $t3, $zero, 1
17    j endif
18 else:
19    addi $t2, $t2, -1
20    add $t3, $t3, $t3
21 endif:
```

Name	Number	Value
\$zero	0	0
\$at	1	268500992
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	1
\$t1	9	3
\$t2	10	3
\$t3	11	10
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	268500996
\$s1	17	2
\$s2	18	1
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
pc		4194368
hi		0
lo		0

Mars Messages: Run I/O

-- program is finished running (dropped off bottom) --

Clear

-- program is finished running (dropped off bottom) --

- Nhận xét:

- Ở câu lệnh “slt \$t0,\$s2,\$s1”, điều kiện $j<i$ đúng nên thanh ghi $t0=1$
- Khi đến câu lệnh bne, giá trị của thanh ghi $t0$ khác với giá trị của $zero$ nên thực hiện rẽ nhánh đến else
- $y=y-1, z=2*z$; $t1$ giữ nguyên giá trị, $t2=3, t3=10$
=> Đúng với lý thuyết đưa ra

b, $i+j \geq 0$

C:\Users\Admin\OneDrive - Hanoi University of Science and Technology\Documents\lab3_huyen - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

lab3_huyen

```
1 # laboratory exercise 3, assignment 1
2 .data
3 i: .word -7
4 j: .word 5
5 .text
6 la $a0, i # load the address of i
7 lw $a1, 0($a0) # load the value of i to register $a1
8 la $a0, j # load the address of j
9 lw $a2, 0($a0) # load the value of j to register $a2
10 li $t1, 1
11 li $t2, 2
12 li $t3, 3
13 add $a0, $a1, $a2 # $a0 = i + j
14 start:
15 blez $a0, else
16 addi $t1, $t1, 1 # x = x + 1
17 addi $t3, $zero, 1 # z = 1
18 j endif # skip "else" part
19 else:
20 addi $t2, $t2, -1 # y = y - 1
21 add $t3, $t3, $t3 # z = 2 * z
22 endif:
23
```

Line: 23 Column: 1 Show Line Numbers

Mars Messages Run I/O

Clear

-- program is finished running (dropped off bottom) --

-- program is finished running (dropped off bottom) --

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0
\$at	1	268500992
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	1
\$t2	10	1
\$t3	11	6
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	-2
\$s1	17	-7
\$s2	18	5
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
pc		4194368
hi		0
lo		0

8:58:32 PM 9/3/2024

c, $j+j > m+n$

- Khởi tạo với $i=5, j=5, m=4, n=5$

C:\Users\Admin\OneDrive - Hanoi University of Science and Technology\Documents\lab3_huyen - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

lab3_huyen

```
1 # laboratory exercise 3, assignment 1
2 .data
3 i: .word 5
4 j: .word 5
5 .text
6 la $a0, i # load the address of i
7 lw $a1, 0($a0) # load the value of i to register $a1
8 la $a0, j # load the address of j
9 lw $a2, 0($a0) # load the value of j to register $a2
10 li $t1, 1
11 li $t2, 2
12 li $t3, 3
13 add $a0, $a1, $a2 # $a0 = i + j
14 li $m1, 4 # m = 4
15 li $n2, 5 # n = 5
16 add $a3, $a1, $a2 # $a3 = m + n
17 sub $a4, $a0, $a3 # $a4 = i + j - m - n
18 start:
19 blez $a4, else
20 addi $t1, $t1, 1 # x = x + 1
21 addi $t3, $zero, 1 # z = 1
22 j endif # skip "else" part
23 else:
24 addi $t2, $t2, -1 # y = y - 1
25 add $t3, $t3, $t3 # z = 2 * z
26 endif:
27
```

Line: 27 Column: 1 Show Line Numbers

Mars Messages Run I/O

Clear

-- program is finished running (dropped off bottom) --

-- program is finished running (dropped off bottom) --

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0
\$at	1	268500992
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	2
\$t2	10	2
\$t3	11	1
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	10
\$s1	17	4
\$s2	18	5
\$s3	19	9
\$s4	20	1
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
pc		4194384
hi		0
lo		0

8:59:46 PM 9/3/2024

d, $(i \leq j) \text{ AND } (i+j \geq 0)$

- Khởi tạo với $i=5, j=7$

CAUsers\Admin\OneDrive - Hanoi University of Science and Technology\Documents\lab3_huyen - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

lab3_huyen

```
3 j: .word 7
4 .text
5 la $a0, i # load the address of i
6 lw $s1, 0($a0) # load the value of i to register $s1
7 la $a0, j # load the address of j
8 lw $s2, 0($a0) # load the value of j to register $s2
9 li $t1, 1 # x
10 li $t2, 2 # y
11 li $t3, 3 # z
12 add $s0, $s1, $s2 # Tính i + j và lưu vào $s0
13 ble $s1, $s2, check1 # Nếu i <= j, nhảy đến check1
14 bgez $s0, check2 # Nếu i + j >= 0, nhảy đến check2
15 j else_block # Nếu không thỏa mãn điều kiện, nhảy đến else_block
16 check1:
17 bgez $s0, if_block # Nếu i + j >= 0, nhảy đến if_block
18 j else_block # Nếu không thỏa mãn điều kiện, nhảy đến else_block
19 check2:
20 ble $s1, $s2, if_block # Nếu i <= j, nhảy đến if_block
21 j else_block # Nếu không thỏa mãn điều kiện, nhảy đến else_block
22 if_block:
23 addi $t1, $t1, 1 # x = x + 1
24 addi $t3, $zero, 1 # z = 1
25 j end_block # Nhảy đến end_block để kết thúc chương trình
26 else_block:
27 addi $t2, $t2, -1 # y = y - 1
28 add $t3, $t3, $t3 # z = 2 * z
29 end_block:
```

Line: 25 Column: 58 ☒ Show Line Numbers

Mars Messages Run I/O

Clear

-- program is finished running (dropped off bottom) --

-- program is finished running (dropped off bottom) --

Registers Coproc 1 Coproc 0

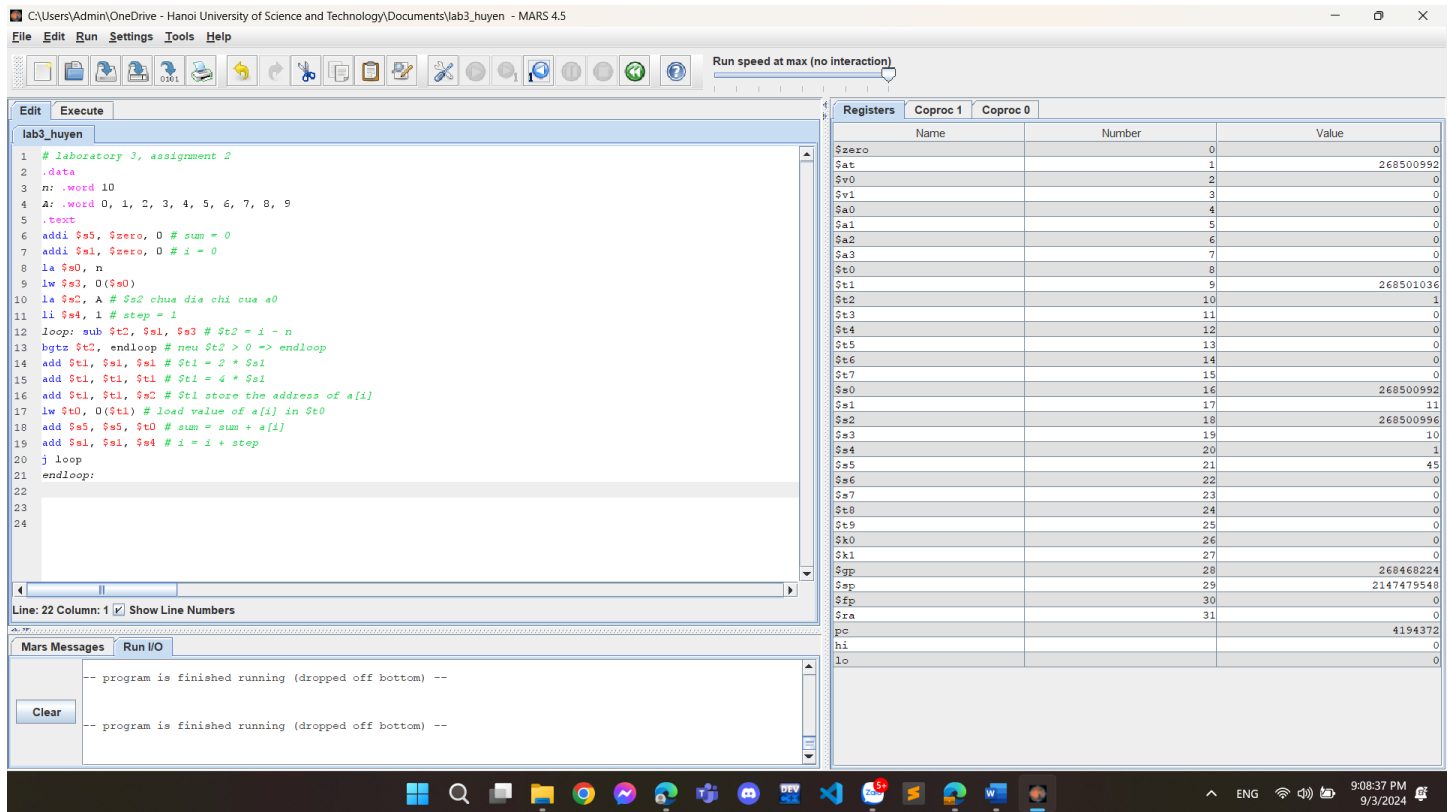
Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	2
\$t2	10	2
\$t3	11	1
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	12
\$s1	17	5
\$s2	18	7
\$s3	19	0
\$s4	20	0
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268460224
\$ap	29	2147479548
\$fp	30	0
\$ra	31	0
pc		4194400
hi		0
lo		0

9:01:11 PM 9/3/2024

Assignment 2

Khởi tạo mảng A: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

a, i <= n



- Nhận xét:

- Đoạn code trên thực hiện tính tổng các phần tử trong mảng arr

- Câu lệnh ‘add \$s1, \$s1, \$s4’ để tăng chỉ số của mảng theo step
- Câu lệnh ‘add \$t1, \$s1, \$s1’ và ‘add \$t1, \$t1, \$t1’ được sử dụng để tính 4*i, vì các giá trị của mảng arr có độ dài 4 bytes.
- Câu lệnh ‘add \$t1, \$t1, \$s2’ câu lệnh này được sử dụng để lấy địa chỉ của phần tử thứ i ở trong mảng và lưu vào t1. Phải sử dụng câu lệnh này vì ta sẽ truy cập vào các phần tử trong mảng dựa trên địa chỉ của các phần tử.
- Câu lệnh ‘lw \$t0, 0(\$t1)’ lưu giá trị của phần tử thứ i trong mảng vào thanh ghi \$t0.
- Câu lệnh ‘add \$s5, \$s5, \$t0’ cộng dần các giá trị các phần tử trong mảng vào sum.
- Câu lệnh ‘bne \$s1, \$s3, loop’ là điều kiện vòng lặp so sánh với n thì sẽ tiến hành rẽ nhánh về nhãn loop, nếu không, thì không thực hiện rẽ nhánh.

- Kết quả: Kết quả thanh ghi \$s5: sum = 45

=> Đúng với lý thuyết đưa ra

b, sum >= 100

C:\Users\Admin\OneDrive - Hanoi University of Science and Technology\Documents\lab3_huyen - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

lab3_huyen

```

1 .data
2 n: .word 10
3 A: .word 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
4 .text
5 addi $a5, $zero, 0 # sum = 0
6 addi $a1, $zero, 0 # i = 0
7 la $n0, n
8 lw $a3, 0($n0)
9 la $a2, A # $a2 chua dia chi cua a0
10 li $a4, 1 # step = 1
11 loop:
12 add $t1, $a1, $a1 # $t1 = 2 * $a1
13 add $t1, $t1, $t1 # $t1 = 4 * $a1
14 add $t1, $t1, $a2 # $t1 store the address of a[i]
15 lw $t0, 0($t1) # load value of a[i] in $t0
16 add $a5, $a5, $t0 # sum = sum + a[i]
17 bge $a5, 100, endloop # neu $a5 >= 100 => endloop
18 add $a1, $a1, $a4 # i = i + step
19 bge $a1, $a3, endloop # neu i >= n => endloop
20 j loop
21 endloop:
22
23
24
25

```

Line: 22 Column: 1 ☒ Show Line Numbers

Mars Messages Run I/O

-- program is finished running (dropped off bottom) --

Clear

-- program is finished running (dropped off bottom) --

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	9
\$t1	9	268501032
\$t2	10	0
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	268500592
\$s1	17	10
\$s2	18	268500596
\$s3	19	10
\$s4	20	1
\$s5	21	45
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	214747548
\$fp	30	0
\$ra	31	0
pc		4194380
hi		0
lo		0

9:12:57 PM 9/3/2024

c, A[i] % 2 = 0

C:\Users\Admin\OneDrive - Hanoi University of Science and Technology\Documents\lab3_huyen - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

lab3_huyen

```

1 .data
2 n: .word 10
3 A: .word 1, 1, 2, 3, 4, 5, 6, 7, 8, 9
4 .text
5 addi $a5, $zero, 0 # sum = 0
6 addi $a1, $zero, 0 # i = 0
7 la $n0, n
8 lw $a3, 0($n0)
9 la $a2, A # $a2 chua dia chi cua a0
10 li $a4, 1 # step = 1
11 li $n7, 2
12 loop:
13 add $t1, $a1, $a1 # $t1 = 2 * $a1
14 add $t1, $t1, $t1 # $t1 = 4 * $a1
15 add $t1, $t1, $a2 # $t1 store the address of a[i]
16 lw $t0, 0($t1) # load value of a[i] in $t0
17 rem $t8, $t0, 2
18 beqz $t8, endloop
19 add $a5, $a5, $t0 # sum = sum + a[i]
20 add $a1, $a1, $a4 # i = i + step
21 j loop
22 endloop:
23
24
25
26

```

Line: 23 Column: 1 ☒ Show Line Numbers

Mars Messages Run I/O

-- program is finished running (dropped off bottom) --

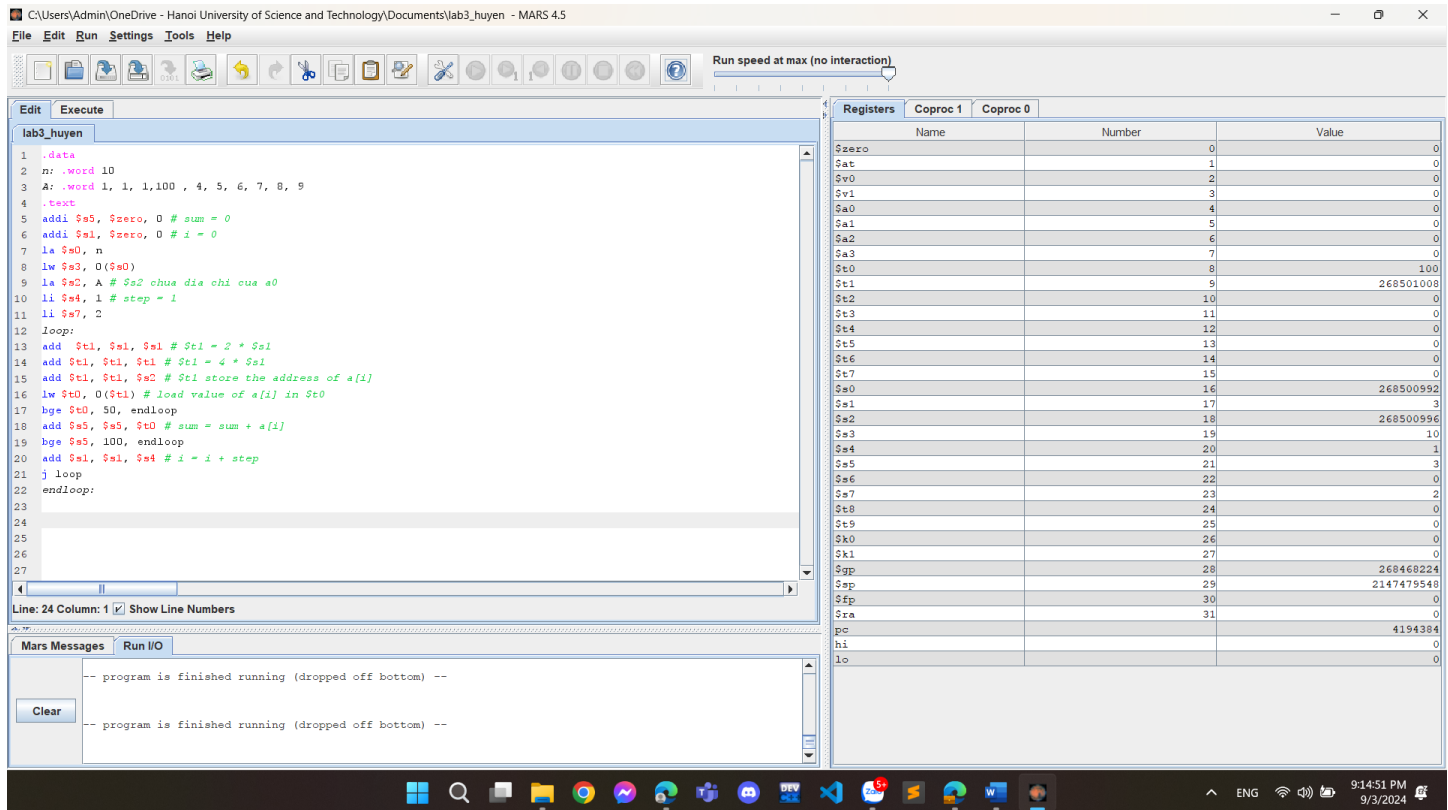
Clear

-- program is finished running (dropped off bottom) --

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0
\$at	1	2
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	2
\$t1	9	268501004
\$t2	10	0
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	268500592
\$s1	17	2
\$s2	18	268500596
\$s3	19	10
\$s4	20	1
\$s5	21	2
\$s6	22	0
\$s7	23	2
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	214747548
\$fp	30	0
\$ra	31	0
pc		4194384
hi		0
lo		1

9:13:51 PM 9/3/2024

d, sum >= 100 OR A[i] >= 50



Assignment 3

Sample Program 1:

- Các thanh ghi bị ảnh hưởng bởi các lệnh jump/branch:
 - \$t0, \$t1, \$t2, \$t3: các thanh ghi này được thay đổi dựa trên các điều kiện và nhánh được thực hiện trong chương trình.
- Giải thích về cách tính toán địa chỉ đích:
 - j: lệnh j thực hiện một nhảy vô điều kiện, địa chỉ đích là địa chỉ được chỉ định trong lệnh.
 - bne: lệnh bne nhảy nếu hai toán hạng không bằng nhau. Nếu điều kiện đúng, sẽ xảy ra nhánh; ngược lại, lệnh tiếp theo được thực hiện. Địa chỉ đích là địa chỉ hiện tại cộng với độ lệch được chỉ định trong lệnh.
 - else: trong trường hợp này, else là một nhãn, và địa chỉ đích của lệnh bne là địa chỉ của nhãn else.

Sample Program 2:

- Các thanh ghi bị ảnh hưởng bởi các lệnh jump/branch:

\$s1, \$s4, \$s5, \$t0, \$t1: các thanh ghi này được thay đổi trong quá trình thực thi của chương trình.

- Giải thích về cách tính toán địa chỉ đích:
 - bne: tương tự như Chương trình Mẫu 1, lệnh bne nhảy nếu hai toán hạng không bằng nhau. Địa chỉ đích được tính dựa trên địa chỉ hiện tại cộng với độ lệch được chỉ định trong lệnh.
 - loop: địa chỉ đích của lệnh bne là địa chỉ của nhãn loop.

Sample Program 3:

- Các thanh ghi bị ảnh hưởng bởi các lệnh jump/branch:
 - \$s0, \$s1, \$s2, \$s3, \$t0, \$t1, \$t2: các thanh ghi này được thay đổi trong quá trình thực thi của chương trình.
- Giải thích về cách tính toán địa chỉ đích:
 - beq: lệnh beq nhảy nếu hai toán hạng bằng nhau. Địa chỉ đích được tính dựa trên địa chỉ hiện tại cộng với độ lệch được chỉ định trong lệnh.
 - case_0, case_1, case_2, default: đây là các nhãn được sử dụng để nhánh. Các địa chỉ đích của các lệnh beq là địa chỉ của những nhãn này.

Assignment 4

- Mã nguồn

```
.data
```

```
arr: .word
```

```
#Generate a random number in the range of [0, 100]
```

```
.text
```

```
la $s0, arr #load address of arr into $s0
```

```
addi $s1,$zero,0 #count i
```

```
addi $s2,$zero, 40 #arr with 40 elements
```

```
loop_1: #initialize elements for arr
```

```
slt $t0, $s1, $s2 # i < n?
```

```
beq $t0 $zero, endloop_1 #branch to endloop if i >= n
```

```
add $s3,$s1,$s1 #s3=2*s1
```

```
add $s3,$s3,$s3 #s3=4*s1
```

```
add $s4,$s0,$s3 #s4 stores the address of A[i]
```

```
li $v0, 42 # System call code to generate random integer
```

```
li $a1, 200 # The upper bound is set in $a1
```

```
syscall # The random number is stored in $a0
```


sw \$a0, 0(\$s4) #stores value of A[i] in \$a0 register into address memory, that was saved in \$s4

addi \$s1,\$s1,1 #i = i + 1(step)

j loop_1

endloop_1:

addi \$s1,\$zero,0 #load value 0 into \$s1 (reset count i = 0)

#largest element in \$t1 and address in \$t4

#second largest element in \$t2 address in \$t5

#third largest element in \$t3 address in \$t6

loop_2: #find 3 largest elements in arr

slt \$t0, \$s1, \$s2 # i < n?

beq \$t0,\$zero, endloop_2 #brand to endloop if i >= n

add \$s3,\$s1,\$s1 #s3=2*s1

add \$s3,\$s3,\$s3 #s3=4*s1

add \$s4,\$s0,\$s3 #s4 stores the address of A[i]

lw \$s5, 0(\$s4) #load value of A[i] in \$s5

slt \$s6, \$t1, \$s5 #largest number now < A[i]?

beq \$s6,\$zero, else_1

add \$t1,\$zero,\$s5 #update largest number

add \$t4, \$zero, \$s4 #update address of largest number

j endif

else_1:

slt \$s6, \$t2, \$s5 #second largest number now < A[i]?

beq \$s6,\$zero, else_2

add \$t2,\$zero,\$s5 #update second largest number

add \$t5, \$zero, \$s4 # #update address of second largest number

j endif

else_2:

slt \$s6, \$t3, \$s5 #third largest number now < A[i]?

beq \$s6,\$zero, endif

add \$t3,\$zero,\$s5 #update third largest number

add \$t6, \$zero, \$s4 #update address of third largest number

endif:

addi \$s1, \$s1, 1

j loop_2

endloop_2:

```

li $v0, 1 # System call code to print an integer on the console
add $a0,$t1,$zero #$a0 = largest number
syscall # The integer needs to be printed out is stored in $a0
add $a0, $zero, 10 #ma ascii de xuong dong
li $v0,11
syscall

```

```

li $v0, 1
add $a0,$t4, $zero #$a0 = address of largest number
syscall # The integer needs to be printed out is stored in $a0
add $a0, $zero, 10 #ma ascii de xuong dong
li $v0,11
syscall

```

```

li $v0, 1
add $a0,$t2, $zero #$a0 = second largest number
syscall # The integer needs to be printed out is stored in $a0
add $a0, $zero, 10 #ma ascii de xuong dong
li $v0,11
syscall

```

```

li $v0, 1
add $a0,$t5, $zero #$a0 = address of second largest
syscall # The integer needs to be printed out is stored in $a0
add $a0, $zero, 10 #ma ascii de xuong dong
li $v0,11
syscall

```

```

li $v0, 1
add $a0,$t3, $zero #$a0 = third largest number
syscall # The integer needs to be printed out is stored in $a0
add $a0, $zero, 10 #ma ascii de xuong dong
li $v0,11
syscall

```

```

li $v0, 1
add $a0,$t6, $zero #$a0 = address of third largest
syscall # The integer needs to be printed out is stored in $a0

```

- Giải thích:

- la \$s0, arr: Load địa chỉ của mảng vào thanh ghi \$s0.
- addi \$s1, %zero, 0: Khởi tạo biến i (vị trí trong mảng) bằng 0.
- addi \$s2, zero, 40: Gán giá trị 40 cho n (số phần tử của mảng).
- loop_1: Khởi tạo giá trị cho các phần tử của mảng.

- Lệnh `slt $t0, $s1, $s2` kiểm tra xem $i < n$ hay không.
- Nếu không (`beq %t0, $zero, endloop_1`) nhảy đến nhãn `endloop_1`.
- Cập nhật địa chỉ của phần tử tiếp theo trong mảng và lấy một số ngẫu nhiên từ 0 đến 199 để lưu vào địa chỉ đó.
- Tăng `i` lên 1 và quay lại `loop_1`
- `loop_2`: Tìm ba số lớn nhất và địa chỉ của chúng trong mảng
 - Lệnh `slt $t0, $s1, $s2` kiểm tra xem $i < n$ hay không.
 - Nếu không (`beq %t0, $zero, endloop_2`) nhảy đến nhãn `endloop_2`.
 - Đọc giá trị của phần tử tại vị trí `i` trong mảng.
 - So sánh giá trị này với ba số lớn nhất hiện tại và cập nhật chúng nếu cần.
- `endif`: Tiếp tục tăng `i` lên 1 và quay lại `loop_2`.
- `endloop_2`: In ra ba số lớn nhất và địa chỉ của chúng.

- Kết quả:

C:\Users\Admin\OneDrive - Hanoi University of Science and Technology\Documents\lab3_huyen - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Text Segment

Bkpt	Address	Code	Basic	Source
0x00400000	0x3c011001	lui \$1,4097		7: la \$s0, arr \$load address of arr into \$s0
0x00400004	0x34300000	ori \$16,\$1,0		
0x00400008	0x20110000	addi \$17,\$0,0		8: addi \$s1,\$zero,0 #count i
0x0040000c	0x20120028	addi \$18,\$0,40		9: addi \$s2,\$zero,40 #arr with 40 elements
0x00400010	0x0232402a	slt \$8,\$17,\$18		12: slt \$t0, \$s1, \$s2 # i < n?
0x00400014	0x11000009	beq \$8,\$0,9		13: beq \$t0 \$zero, endloop_1 #branch to endloop if i >= n
0x00400018	0x02319820	add \$19,\$17,\$17		14: add \$s3,\$s1,\$s1 #s3=2*\$s1
0x0040001c	0x02739820	add \$19,\$19,\$19		15: add \$s3,\$s3,\$s3 #s3=4*\$s1
0x00400020	0x0213a020	add \$20,\$16,\$19		16: add \$s4,\$s0,\$s3 #s4 stores the address of A[i]
0x00400024	0x2402002a	addiu \$2,\$0,42		17: li \$v0, 42 # System call code to generate random integer
0x00400028	0x240500c8	addiu \$5,\$0,200		18: li \$a1, 200 # The upper bound is set in \$a1
0x0040002c	0x0000000c	syscall		19: syscall # The random number is stored in \$a0

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	4	94	144	185	153	143	190	35
0x10010020	142	155	139	147	184	173	115	107
0x10010040	167	197	194	108	88	81	90	3
0x10010060	38	76	136	121	138	47	75	83
0x10010080	171	135	80	173	110	123	141	98
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0

Registers

Name	Number	Value
\$zero	0	0
\$at	1	268500992
\$v0	2	1
\$v1	3	0
\$a0	4	268501044
\$a1	5	200
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	197
\$t2	10	194
\$t3	11	173
\$t4	12	268501060
\$t5	13	268501064
\$t6	14	268501044
\$t7	15	0
\$a0	16	268500992
\$s1	17	40
\$s2	18	40
\$s3	19	156
\$s4	20	268501148
\$s5	21	98
\$s6	22	0
\$s7	23	0
\$s8	24	0
\$s9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
\$PC		4194588
\$hi		0
\$lo		0

Mars Messages

174
268501064
173
268501044
-- program is finished running (dropped off bottom) --

