

# Báo cáo thực hành Lab5

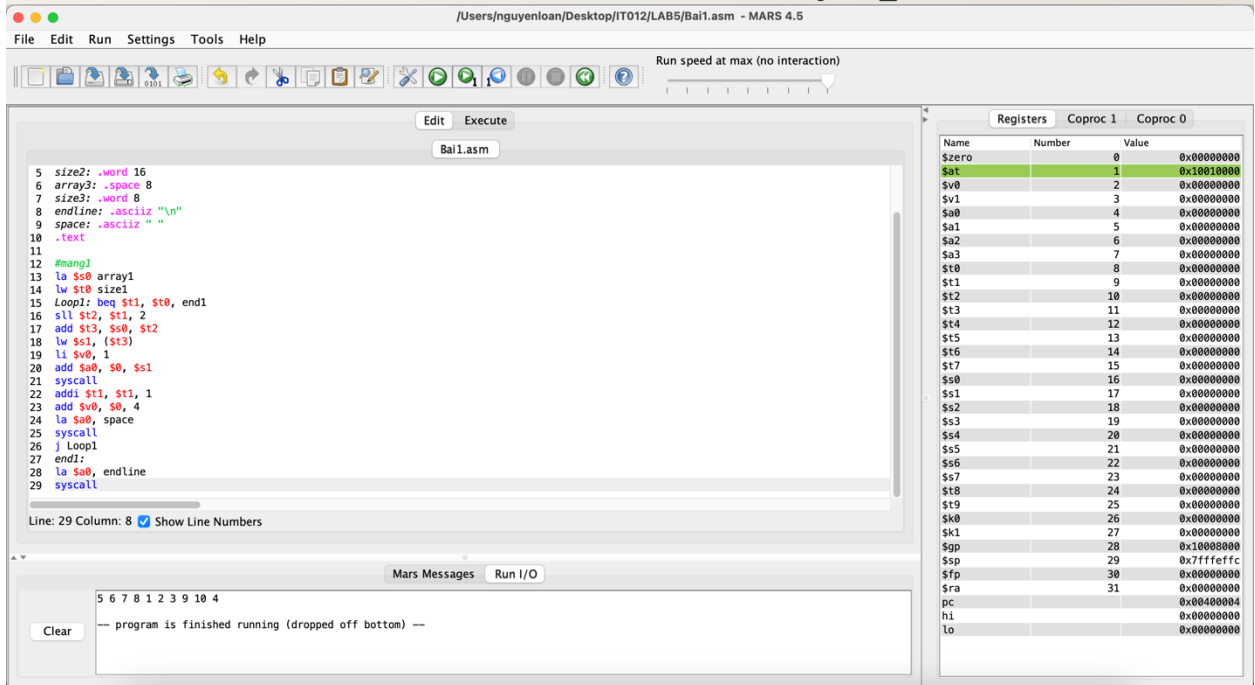
## Môn: IT012

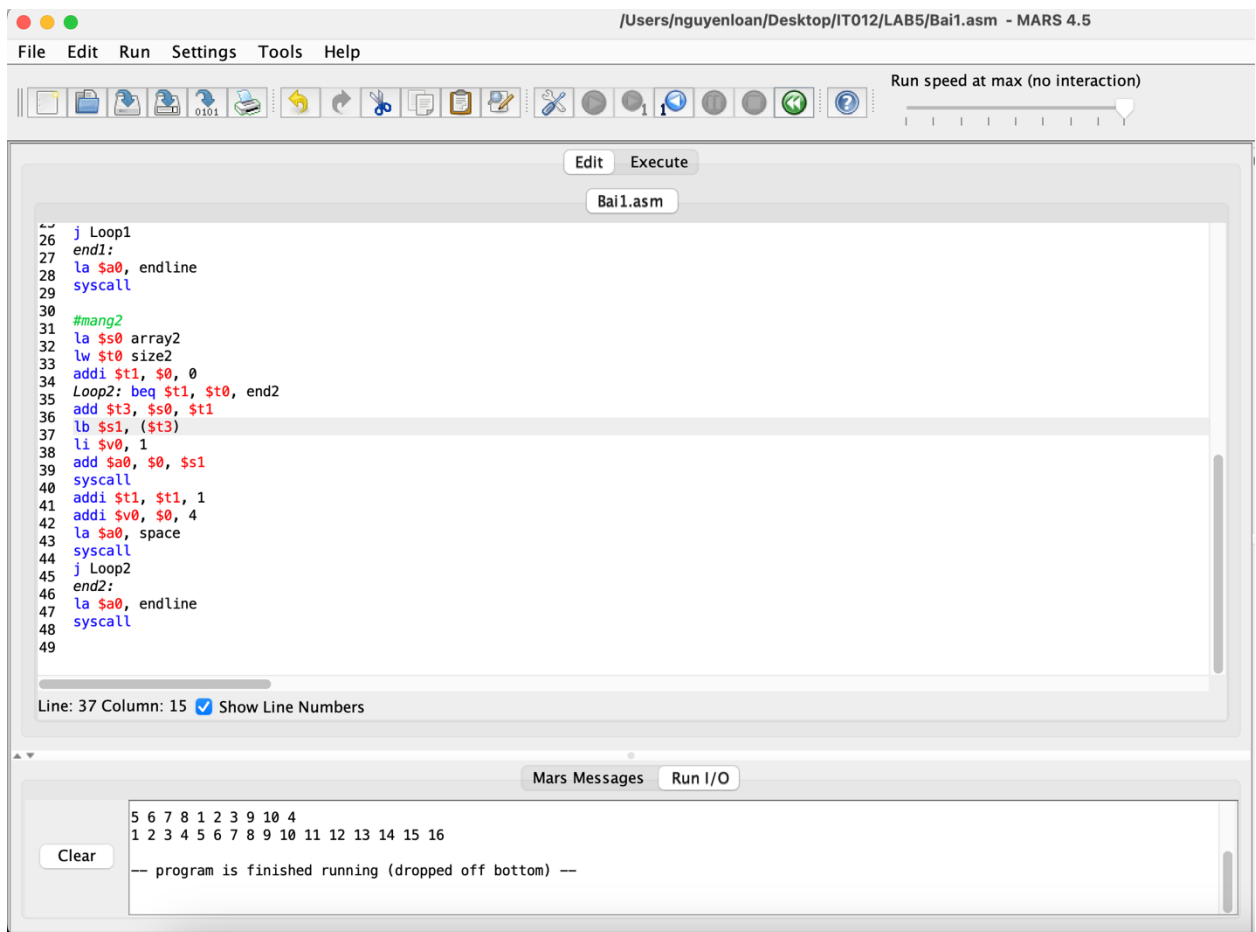
### GVHD: Phan Đình Duy

Họ và tên: Đỗ Phương Duy  
MSSV: 23520362

#### 1. Thao tác mảng:

In ra cửa sổ I/O của MARS tất cả các phần tử của mảng array1 và array2





Gán các giá trị cho mảng array3 sao cho  $\text{array3}[i] = \text{array2}[i] + \text{array2}[\text{size2} - 1 - i]$

```

.text
lw $t0 size3
lw $s0 size2
la $s1 array2
la $s2 array3
Loop:
beq $t1, $t0, end
add $t2, $t1, $s1
lb $t3 ($t2)

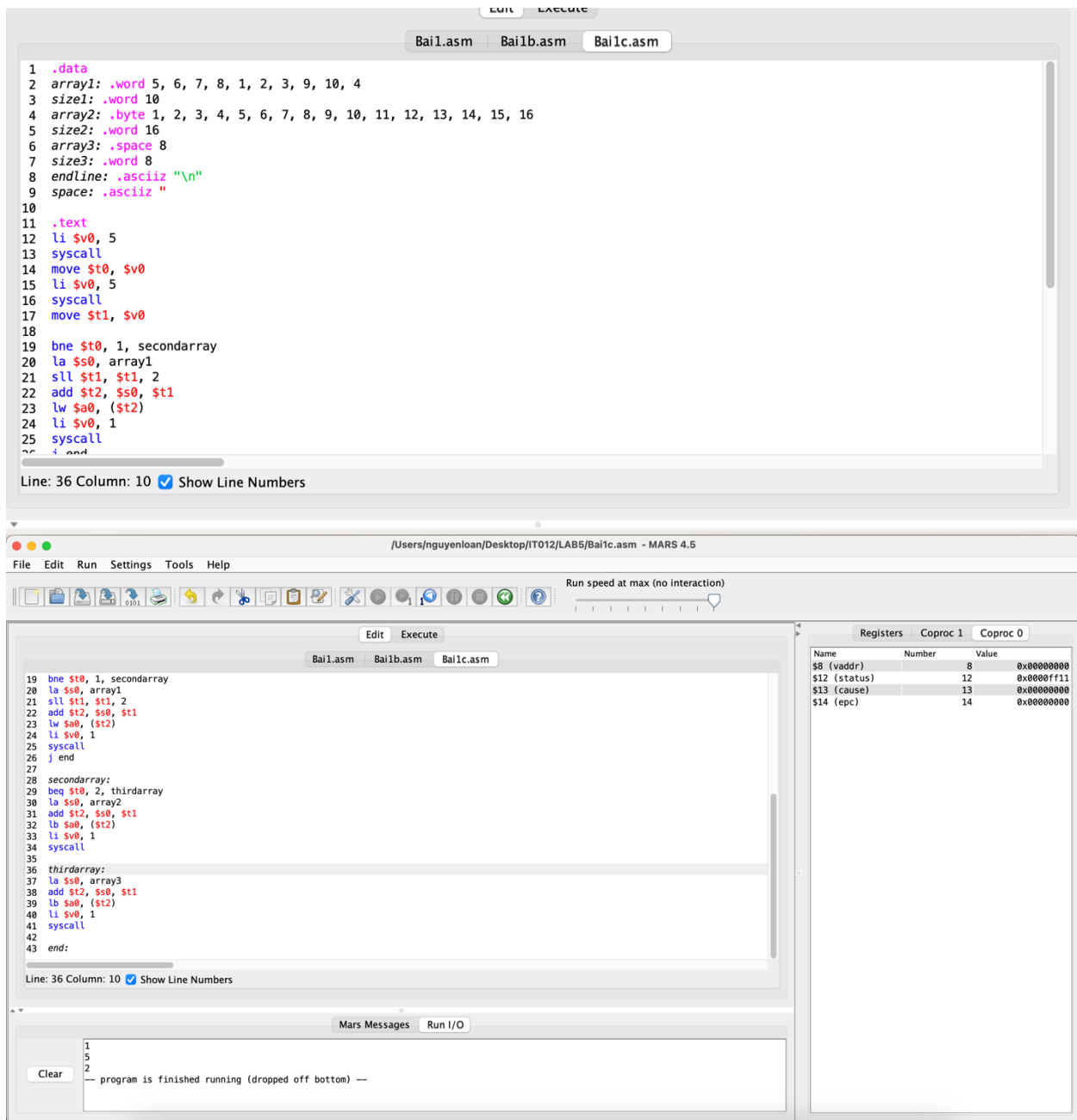
sub $t2, $s0, $t1
subi $t2, $t2, 1
add $t2, $t2, $s1
lb $t4 ($t2)

add $t5, $t3, $t4
add $t6, $t1, $s2
sb $t5, ($t6)
addi $t1, $t1, 1
j Loop
end:

```

Data Segment								
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000005	0x00000006	0x00000007	0x00000008	0x00000001	0x00000002	0x00000003	0x00000009
0x10010020	0x0000000a	0x00000004	0x0000000a	0x04030201	0x08070605	0x0c0b0a09	0x100f0e0d	0x00000010
0x10010040	0x11111111	0x11111111	0x00000008	0x0000000a	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Người sử dụng nhập vào mảng thứ mấy và chỉ số phần tử cần lấy trong mảng đó, chương trình xuất ra phần tử tương ứng.

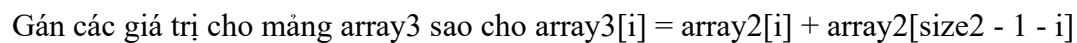


## 2. Thao tác con trỏ:

Con trỏ là một biến lưu địa chỉ của một biến khác. Thao tác với con trỏ trong MIPS là thao tác trực tiếp với địa chỉ bộ nhớ.

Thực hiện lại các yêu cầu của nội dung 1 với con trỏ.

In ra cửa sổ I/O của MARS tất cả các phần tử của mảng array1 và array2



Edit Execute

Bai1.asm
Bai1b.asm
Bai1c.asm
bai2a.asm
bai2b

```

7  size3: .word 8
8  newline: .asciiz "\n"
9  space: .asciiz " "
10
11  .text
12  lw $t0 size3
13  lw $s0 size2
14  la $s1 array2
15  la $s2 array3
16  Loop:
17  beq $t1, $t0, end
18  lb $t3, ($s1)
19  addi $s1, $s1, 1
20
21  sub $t2, $s0, $t1
22  subi $t2, $t2, 1
23  add $t2, $t2, $s1
24  lb $t4, ($t2)
25
26  add $t5, $t3, $t4
27  sb $t5, ($s2)
28  addi $s2, $s2, 1
29  addi $t1, $t1, 1
30  j Loop
31  end:

```

Line: 27 Column: 15 ☒ Show Line Numbers

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000005	0x00000006	0x00000007	0x00000008	0x00000001	0x00000002	0x00000003	0x00000009
0x10010020	0x0000000a	0x00000004	0x0000000a	0x04030201	0x08070605	0x0c0b0a09	0x100f0e0d	0x00000010
0x10010040	0x11111111	0x11111111	0x00000008	0x0000000a	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

← → 0x10010000 (.data) ⌵
☒ Hexadecimal Addresses
☒ Hexadecimal Values
☐ ASCII

Người sử dụng nhập vào mảng thứ mấy và chỉ số phần tử cần lấy trong mảng đó, chương trình xuất ra phần tử tương ứng.

EditExecute

Bai1.asmBai1b.asmBai1c.asmbai2a.asmbai2bbai2c.asm

```
1 .data
2 array1: .word 5, 6, 7, 8, 1, 2, 3, 9, 10, 4
3 size1: .word 10
4 array2: .byte 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
5 size2: .word 16
6 array3: .space 8
7 size3: .word 8
8 endLine: .asciiz "\n"
9 space: .asciiz " "
10
11 .text
12 li $v0, 5
13 syscall
14 move $t0, $v0
15 li $v0, 5
16 syscall
17 move $t1, $v0
18
19 bne $t0, 1, secondarray
20 la $s0, array1
21 sll $t1, $t1, 2
22 add $t2, $s0, $t1
23 lw $a0, ($t2)
24 li $v0, 1
25 syscall
26 j end
```

Line: 16 Column: 9 ✓ Show Line Numbers

EditExecute

Bai1.asmBai1b.asmBai1c.asmbai2a.asmbai2bbai2c.asm

```
19 bne $t0, 1, secondarray
20 la $s0, array1
21 sll $t1, $t1, 2
22 add $t2, $s0, $t1
23 lw $a0, ($t2)
24 li $v0, 1
25 syscall
26 j end
27
28 secondarray:
29 beq $t0, 2, thirdarray
30 la $s0, array2
31 add $t2, $s0, $t1
32 lb $a0, ($t2)
33 li $v0, 1
34 syscall
35
36 thirdarray:
37 la $s0, array3
38 add $t2, $s0, $t1
39 lb $a0, ($t2)
40 li $v0, 1
41 syscall
42
43 end:
```

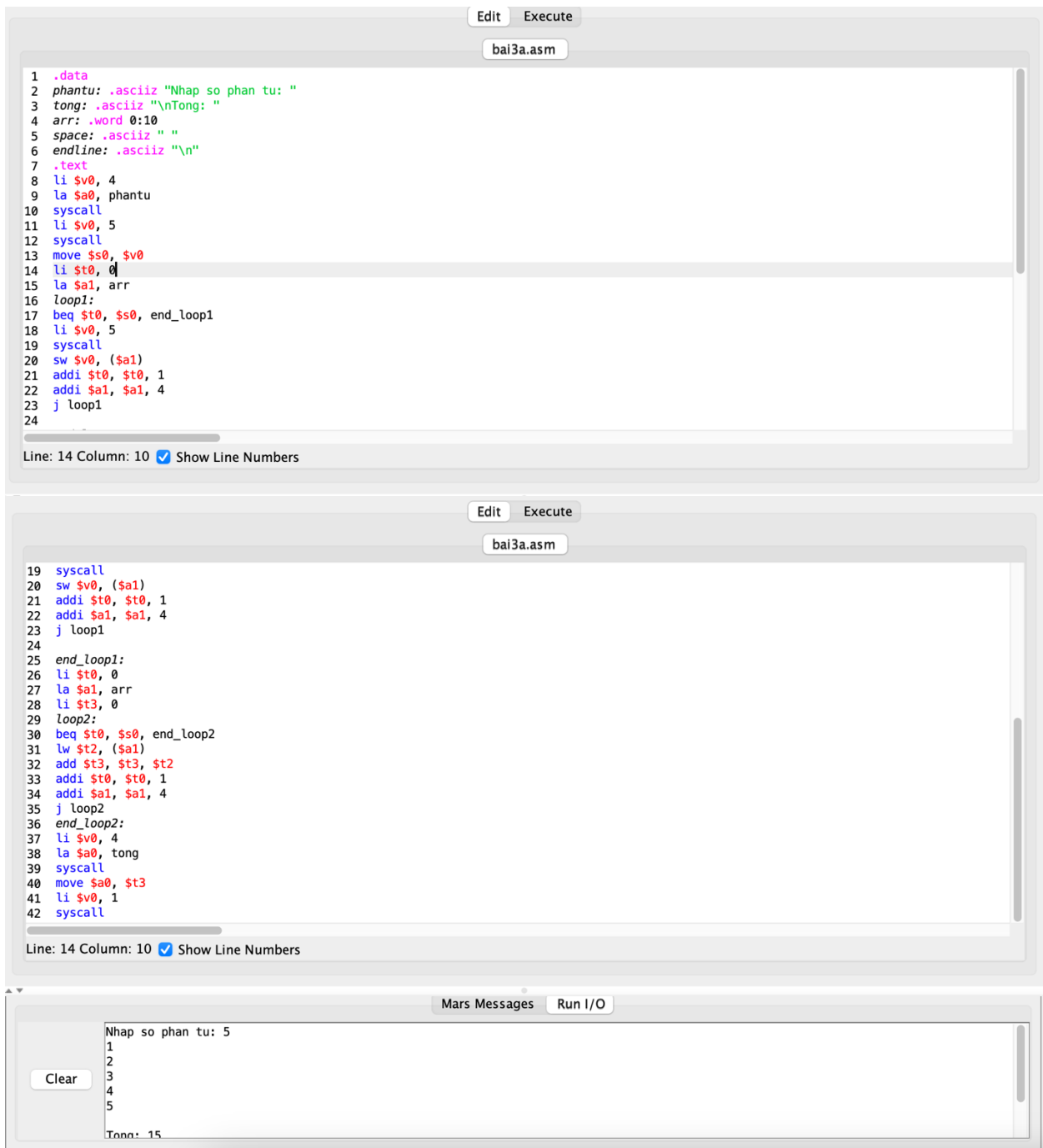
Line: 16 Column: 9 ✓ Show Line Numbers

### 3. Bài tập: (chỉ sử dụng con trỏ):

a. Nhập một mảng các số nguyên n phần tử (nhập vào số phần tử và giá trị của từng phần tử), xuất ra cửa sổ I/O của MARS theo từng yêu cầu sau:

✓ Xuất ra giá trị lớn nhất và nhỏ nhất của mảng

✓ Tổng tất cả các phần tử của mảng:



✓ Người sử dụng nhập vào chỉ số của một phần tử nào đó và giá trị của phần tử đó được in ra cửa sổ.

**b. Nhập một mảng các số nguyên n phần tử (nhập vào số phần tử và giá trị của từng phần tử). Mảng này gọi là A.**

Chuyển dòng lệnh C dưới đây sang mã assembly của MIPS. Với các biến nguyên `i, j` được gán lần lượt vào thanh ghi `$s0, $s1`; và địa chỉ nền của mảng số nguyên A được lưu trong thanh ghi `$s3`



if (i<j) A[i]= i;  
else A[i] = j;

```
.text

slt $t0, $s0, $s1
beq $t0, 0, else

sll $t1, $s0, 2
add $s3, $s3, $t1
sw $s0, ($s3)

else:
sll $t1, $s0, 2
add $s3, $s3, $t1
sw $s1, ($s3)
```