

1. What is the usage of \$zero? What happens if you execute `addi $zero, $zero, 5` ? (5%)

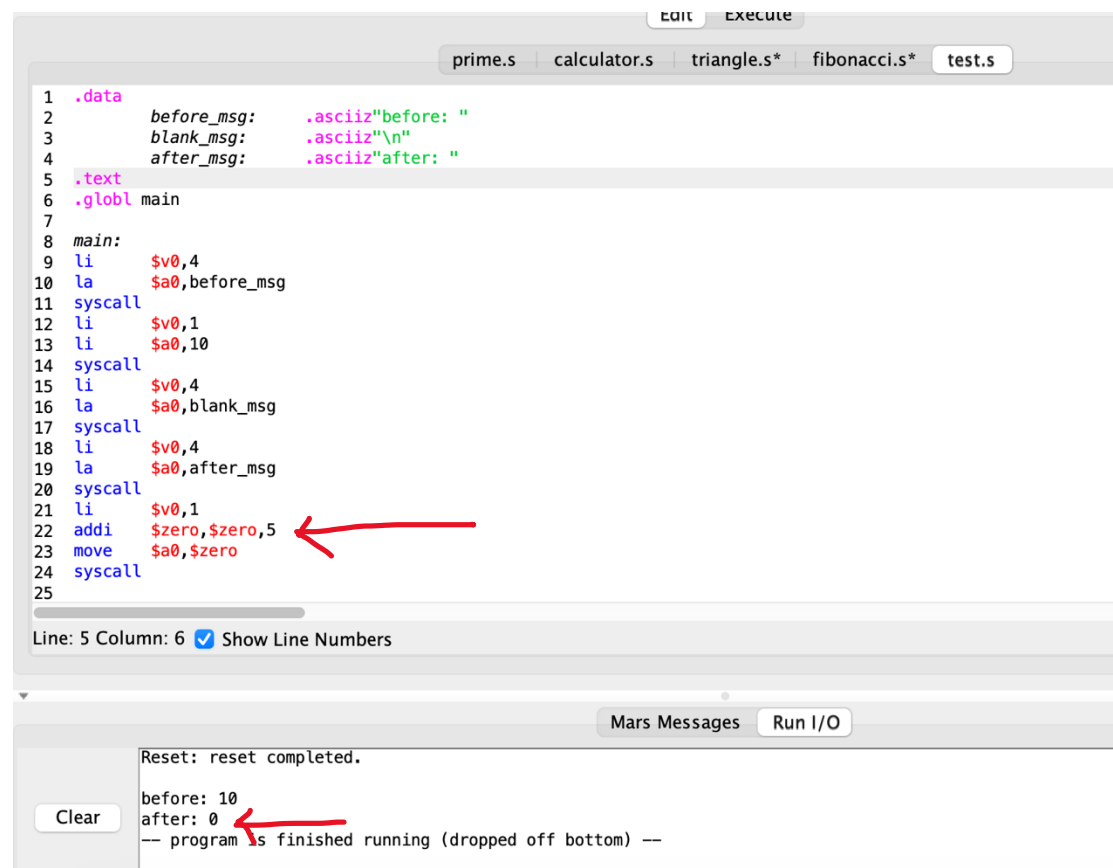
\$zero 常用來執行二進制的 NOT 運算

例如

`$a0=0x00000006`

執行 `nor $s0,$a0,$zero`

則 `$s0=0x11111119`



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

```
1 .data
2     before_msg: .asciiz"before: "
3     blank_msg: .asciiz"\n"
4     after_msg: .asciiz"after: "
5 .text
6 .globl main
7
8 main:
9     li $v0,4
10    la $a0,before_msg
11    syscall
12    li $v0,1
13    li $a0,10
14    syscall
15    li $v0,4
16    la $a0,blank_msg
17    syscall
18    li $v0,4
19    la $a0,after_msg
20    syscall
21    li $v0,1
22    addi $zero,$zero,5
23    move $a0,$zero
24    syscall
25
```

A red arrow points to line 22, `addi $zero,$zero,5`. The status bar at the bottom indicates "Line: 5 Column: 6" and "Show Line Numbers" is checked.

The "Mars Messages" window shows the following output:

```
Reset: reset completed.
before: 10
after: 0
-- program is finished running (dropped off bottom) --
```

A red arrow points to the "after: 0" line in the output.

2. How to use the stack to ensure that the value of each register is correctly saved when executing a recursive function? (5%)

先 push 出要用到的 stack 空間，如果要存 2 個 word，因為每個

word 為 4Bytes，所以 `$sp` 就要-8，當使用完後 `$sp` 要+8 使其復位

MIPS code:

fact:		
	addi \$sp, \$sp, -8	# adjust stack for 2 items
	sw \$ra, 4(\$sp)	# save return address
	sw \$a0, 0(\$sp)	# save argument
	slti \$t0, \$a0, 1	# test for n < 1
	beq \$t0, \$zero, L1	# branch if n >= 1
	addi \$v0, \$zero, 1	# if so, result is 1
	addi \$sp, \$sp, 8	# pop 2 items from stack
	jr \$ra	# and return
L1:	addi \$a0, \$a0, -1	# else decrement n
	jal fact	# recursive call
	lw \$a0, 0(\$sp)	# restore original n
	lw \$ra, 4(\$sp)	# and return address
	addi \$sp, \$sp, 8	# pop 2 items from stack
	mul \$v0, \$a0, \$v0	# multiply to get result
	jr \$ra	# and return

3. What was the most challenging part for you in this homework?
(10%)

因為是第一次寫組合語言，思路跟高階語言不太一樣，作業這幾題寫下來有意識到自己遇到迴圈常常會卡住，要很清楚硬體下一步該幹嘛、暫存器存的值多少以及執行指令的順序，如果用到 jal 指令時要記得 \$ra 存的位址會變動，使用 jr 指令進行 return 時也要確認現在 \$ra 存的位址是否為想要跳回去的位址