1

```
        .data
a:      .word       0xffffffa5
b:      .word       0x000003ab
c:      .space      4

        .text
        .globl      main
main:
. . .
la      $a0, a
la      $a1, b
la      $a3, c
lw      $t0, 0($a0)
lw      $t1, 0($a1)
sll     $t1, $t1, 2
sub     $t2, $t0, $t1
sw      $t2, 0($a3)
...
```

endereço

→ Reservo 4 bytes em memória para o array

$a0 tem o endereço da variável a

meter os endereços no $a0 $a1 $a2

$t0 = &[a0]

→ coloca em $t0 a informação da memória que está no endereço $a0

$t1 = &[a1]

$t1 = $t1 x 4 = 0x000003ab <<2 = 0... 1110 1010 1100 = 0x00000eac

$t2 = $t0 - $t1 =

```
  FFFFFFA5
+ FFFFF154
  FFFFF0F9
```

= -(00000F06 + 1)

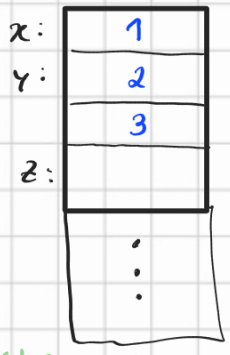
= -(00000F07)

= -(15x16² + 7)

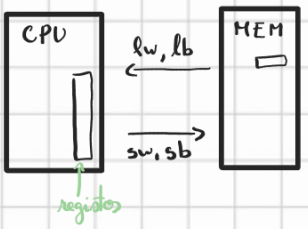
- 0x00000eac = 0xFFFFF154

Se quero fazer coisas como z[i]

```
        .data
x:      .word 1
y:      .word 2,3, ...
z:      .space 8
        .text
```




```
la  $t1, y      $t1 = index
li  $t2, 1      $t2 = 1
sll $t2, $t2, 2 $t2 = 1x4 = 4
add $t2, $t2, $t1 $t2 = index + 4
lw  $s1, 0($t2) $s1 = list[$t2]
```



2. O programa seguinte lê, do teclado, um conjunto de 6 números inteiros e armazena-os no array lista.

```
void main(void)
{
 static int lista[6];
 int i=0;
 printstr( "Insira 6 numeros: ");

 for( i=0; i<6; i++)
 {
       lista[i] = read_int();
 }
 exit();
}
```

&lista[0]	$s0
i	$s1
&lista[i]	$s2
lista[i]	$s3

← alínea b)

a) Traduza o programa para *Assembly* e verifique o seu funcionamento usando a janela do segmento de dados do simulador MARS.

b) Altere o programa anterior (usando o programa da questão 1) de modo a que imprima na consola os números inteiros lidos.

```
.data
result: .asciiz "Insira 6 números: "
lista:  .space 24        # 6x4
.text
.globl main

main: la $s0, lista
      li $s1, 0

      li $v0, 4
      la $a0, result
      syscall

for:  bge $s1, 6, done

      sll $s2, $s1, 2     # i = i x 4
      add $s2, $s2, $s0   # &(lista[i]) = i x 4 + &(lista[0])

      li $v0, 5
      syscall

      sw $v0, 0($s2)      # lista[i] = $v0

      addi $s1, $s1, 1
      j for

done: li $v0, 10
      syscall
```