# KIẾN TRÚC MÁY TÍNH

## WEEK 6

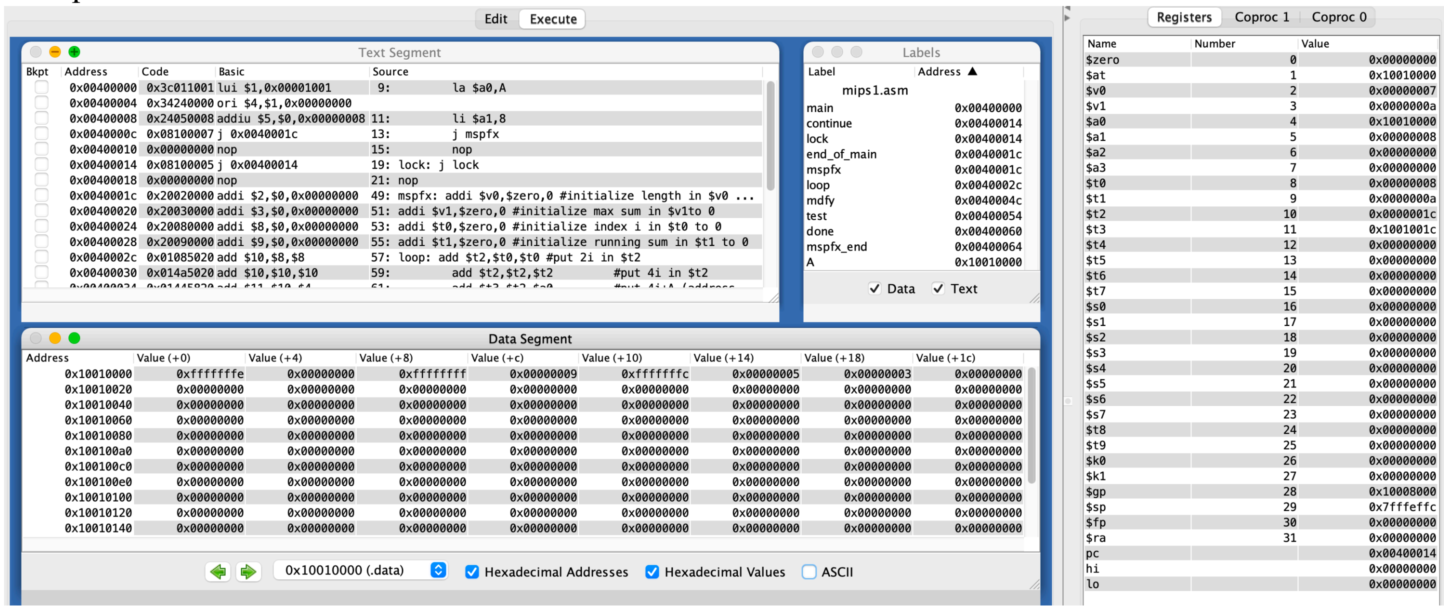
## Assignment 1:

- Chương trình:

```
.data
A: .word -2, 0, -1, 9, -4, 5, 3, 0
.text
main:
	la $a0,A
	li $a1,8
	j mspfx
	nop
continue:
lock: j lock
nop
end_of_main:
#-----------------------------------------------------------------
#Procedure mspfx
# @brief find the maximum-sum prefix in a list of integers
# @param[in] a0 the base address of this list(A) need to be processed
# @param[in] a1 the number of elements in list(A)
# @param[out] v0 the length of sub-array of A in which max sum reachs.
# @param[out] v1 the max sum of a certain sub-array
#-----------------------------------------------------------------
#Procedure mspfx
#function: find the maximum-sum prefix in a list of integers
#the base address of this list(A) in $a0 and the number of
#elements is stored in a1
mspfx: addi $v0,$zero,0 #initialize length in $v0 to 0
addi $v1,$zero,0 #initialize max sum in $v1to 0
addi $t0,$zero,0 #initialize index i in $t0 to 0
addi $t1,$zero,0 #initialize running sum in $t1 to 0
loop: add $t2,$t0,$t0 #put 2i in $t2
	add $t2,$t2,$t2       #put 4i in $t2
	add $t3,$t2,$a0       #put 4i+A (address of A[i]) in $t3
```

```
        lw $t4,0($t3)  #load A[i] from mem(t3) into $t4
        add $t1,$t1,$t4       #add A[i] to running sum in $t1
        slt $t5,$v1,$t1       #set $t5 to 1 if max sum < new sum
        bne $t5,$zero,mdfy #if max sum is less, modify results
        j test #done?
mdfy:
        addi $v0,$t0,1 #new max-sum prefix has length i+1
        addi $v1,$t1,0 #new max sum is the running sum
test: addi $t0,$t0,1 #advance the index i
        slt $t5,$t0,$a1 #set $t5 to 1 if i<n
        bne $t5,$zero,loop #repeat if i<n
done: j continue
mspfx_end:
```

- Kết quả:



```
Text Segment
Bkpt Address    Code       Basic                      Source
     0x00400000 0x3c011001 lui $1,0x00001001          9:          la $a0,A
     0x00400004 0x34240000 ori $4,$1,0x00000000
     0x00400008 0x24050008 addiu $5,$0,0x00000008 11:          li $a1,8
     0x0040000c 0x08100007 j 0x0040001c           13:          j mspfx
     0x00400010 0x00000000 nop                    15:          nop
     0x00400014 0x08100005 j 0x00400014           19: lock: j lock
     0x00400018 0x00000000 nop                    21: nop
     0x0040001c 0x20020000 addi $2,$0,0x00000000  49: mspfx: addi $v0,$zero,0 #initialize length in $v0 ...
     0x00400020 0x20030000 addi $3,$0,0x00000000  51: addi $v1,$zero,0 #initialize max sum in $v1to 0
     0x00400024 0x20080000 addi $8,$0,0x00000000  53: addi $t0,$zero,0 #initialize index i in $t0 to 0
     0x00400028 0x20090000 addi $9,$0,0x00000000  55: addi $t1,$zero,0 #initialize running sum in $t1 to 0
     0x0040002c 0x01085020 add $10,$8,$8          57: loop: add $t2,$t0,$t0 #put 2i in $t2
     0x00400030 0x014a5020 add $10,$10,$10        59:          add $t2,$t2,$t2        #put 4i in $t2
```

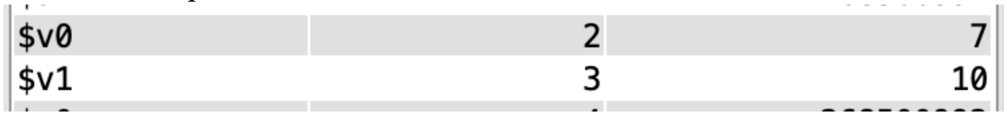
```
Labels
Label           Address
     mips1.asm
main            0x00400000
continue        0x00400014
lock            0x00400014
end_of_main     0x0040001c
mspfx           0x0040001c
loop            0x0040002c
mdfy            0x0040004c
test            0x00400054
done            0x00400060
mspfx_end       0x00400064
A               0x10010000
✓ Data  ✓ Text
```

```
Data Segment
Address     Value (+0)  Value (+4)  Value (+8)  Value (+c)  Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000  0xfffffffe  0x00000000  0xffffffff  0x00000009  0xfffffffc  0x00000005  0x00000003  0x00000000
0x10010020  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000
0x10010040  0x00000000  0x00000000  0x00000000  0x00000000  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
0x100100e0  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000
0x10010100  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000
0x10010120  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000
0x10010140  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000
0x10010000 (.data)   ✓ Hexadecimal Addresses   ✓ Hexadecimal Values   ASCII
```

```
Registers
Name    Number  Value
$zero   0       0x00000000
$at     1       0x10010000
$v0     2       0x00000007
$v1     3       0x0000000a
$a0     4       0x10010000
$a1     5       0x00000008
$a2     6       0x00000000
$a3     7       0x00000000
$t0     8       0x00000008
$t1     9       0x0000000a
$t2     10      0x0000001c
$t3     11      0x1001001c
$t4     12      0x00000000
$t5     13      0x00000000
$t6     14      0x00000000
$t7     15      0x00000000
$s0     16      0x00000000
$s1     17      0x00000000
$s2     18      0x00000000
$s3     19      0x00000000
$s4     20      0x00000000
$s5     21      0x00000000
$s6     22      0x00000000
$s7     23      0x00000000
$t8     24      0x00000000
$t9     25      0x00000000
$k0     26      0x00000000
$k1     27      0x00000000
$gp     28      0x10008000
$sp     29      0x7fffeffc
$fp     30      0x00000000
$ra     31      0x00000000
pc              0x00400014
hi              0x00000000
lo              0x00000000
```

Do dữ liệu đầu vào của chương trình là một mảng A = {-2, 6, -1, 3, -2}

⇨ Ta được kết quả



| | | |
|---|---|---|
| $v0 | 2 | 7 |
| $v1 | 3 | 10 |

Sub lớn nhất là 10 ứng với $v1 và độ dài của mảng đến khi có tổng lớn nhất là 7 ứng với $v0

- Debug từng dòng:

| Step | $pc | Giá trị thanh ghi thay đổi | Ghi chú |
|---|---|---|---|
| 1 | 0x00400004 | $at = 0x10010000 | |
| 2 | 0x00400008 | $a0 = 0x10010000 | $a0 = địa chỉ đầu mảng A |
| 3 | 0x0040000c | $a1 = 0x00000008 | Độ dài mảng A = 8 |
| 4 | 0x004000020 | $v0 = 0x00000000 | Nhảy đến mspfx |

| | | | |
|---|---|---|---|
| 5 | 0x004000024 | $v1 = 0x00000000 | |
| 6 | 0x004000028 | $t0 = 0x00000000 | |
| 7 | 0x00400002c | $t1 = 0x00000000 | |
| 8 | 0x004000030 | $t2 = 0x00000000 | $t2 = 2$t0 |
| 9 | 0x004000030 | $t2 = 0x00000000 | $t2 = 2$t2 = 4$t0 |
| 10 | 0x004000038 | $t3 = 0x10010000 | $t3 = địa chỉ của A[i] |
| 11 | 0x00400003c | $t4 = 0xfffffffe | $t4 = A[i] |
| 12 | 0x004000040 | $t1 = 0xfffffffe | $t1 lưu sum hiện tại |
| 13 | 0x004000044 | $t5 = 0x00000000 | $v1 < $t1 thì $t5 = 1, ngược lại $t5 = 0 |
| 14 | 0x004000048 | | |
| 15 | 0x004000054 | | Nhảy đến test |
| 16 | 0x004000058 | $t0 = 0x00000001 | Tăng chỉ mục lên 1 => xét phần tử tiếp theo |
| … | … | … | … |

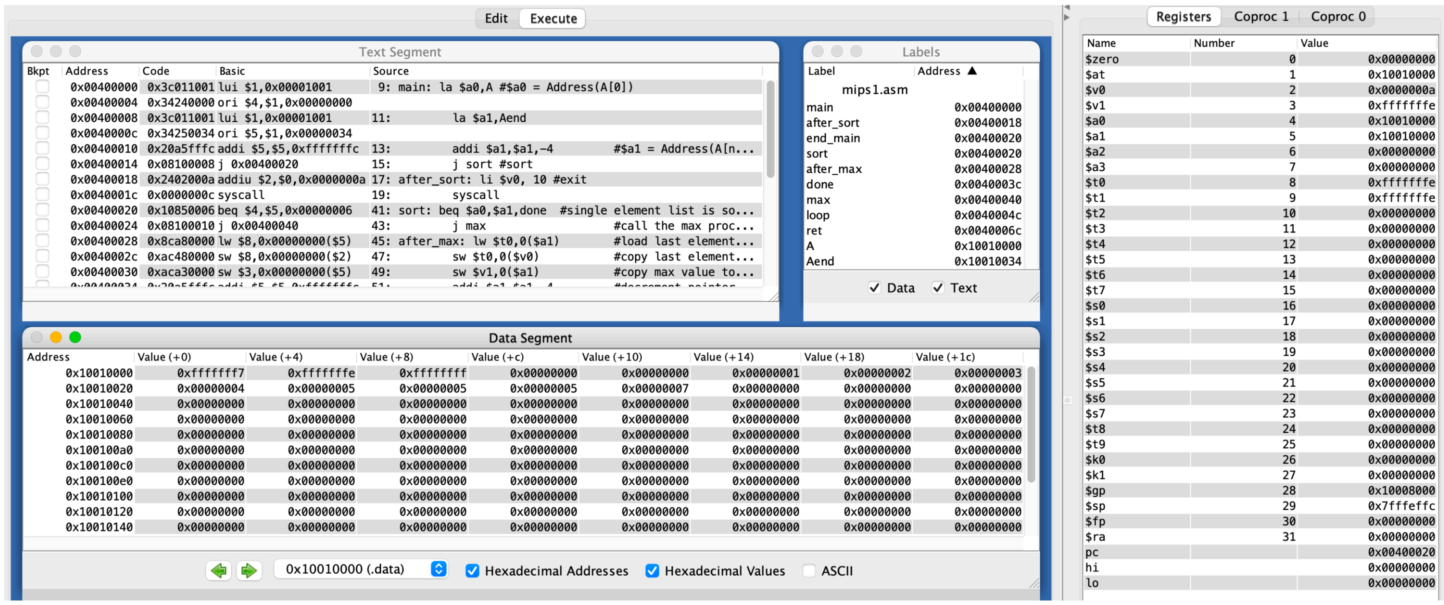
## Assignment 2:

- Chương trình:

```
.data
A: .word 7, -2, 5, 1, 5, 2,0,-1,-9,4,5,3,0
Aend: .word
.text
main: la $a0,A #$a0 = Address(A[0])
       la $a1,Aend
       addi $a1,$a1,-4       #$a1 = Address(A[n-1])
       j sort #sort
after_sort: li $v0, 10 #exit
       syscall
end_main:
#---------------------------------------------------------------
#procedure sort (ascending selection sort using pointer)
#register usage in sort program
#$a0 pointer to the first element in unsorted part
#$a1 pointer to the last element in unsorted part
#$t0 temporary place for value of last element
#$v0 pointer to max element in unsorted part
```

```
#$v1 value of max element in unsorted part
#---------------------------------------------------------------
sort: beq $a0,$a1,done       #single element list is sorted
       j max                 #call the max procedure
after_max: lw $t0,0($a1)     #load last element into $t0
       sw $t0,0($v0)         #copy last element to max location
       sw $v1,0($a1)         #copy max value to last element
       addi $a1,$a1,-4       #decrement pointer to last element
       j sort                #repeat sort for smaller list
done: j after_sort
#-------------------------------------------------------------------------
#Procedure max
#function: fax the value and address of max element in the list
#$a0 pointer to first element
#$a1 pointer to last element
#-------------------------------------------------------------------------
max:
       addi $v0,$a0,0        #init max pointer to first element
       lw $v1,0($v0)         #init max value to first value
       addi $t0,$a0,0        #init next pointer to first
loop:
       beq $t0,$a1,ret        #if next=last, return
       addi $t0,$t0,4         #advance to next element
       lw $t1,0($t0)          #load next element into $t1
       slt $t2,$t1,$v1       #(next)<(max) ?
       bne $t2,$zero,loop    #if (next)<(max), repeat
       addi $v0,$t0,0         #next element is new max element
       addi $v1,$t1,0        #next value is new max value
       j loop                #change completed; now repeat
ret:
       j after_max
```

- Mảng đầu vào là mảng A = {7, -2, 5, 1, 5, 2, 0, -1, -9, 4, 5, 3, 0}
- Kết quả chạy:



```
Bkpt Address    Code       Basic                      Source
     0x00400000 0x3c011001 lui $1,0x00001001           9: main: la $a0,A #$a0 = Address(A[0])
     0x00400004 0x34240000 ori $4,$1,0x00000000
     0x00400008 0x3c011001 lui $1,0x00001001          11:        la $a1,Aend
     0x0040000c 0x34250034 ori $5,$1,0x00000034
     0x00400010 0x20a5fffc addi $5,$5,0xfffffffc     13:        addi $a1,$a1,-4         #$a1 = Address(A[n...
     0x00400014 0x08100008 j 0x00400020               15:        j sort #sort
     0x00400018 0x2402000a addiu $2,$0,0x0000000a     17: after_sort: li $v0, 10 #exit
     0x0040001c 0x0000000c syscall                    19:        syscall
     0x00400020 0x10850006 beq $4,$5,0x00000006       41: sort: beq $a0,$a1,done  #single element list is so...
     0x00400024 0x08100010 j 0x00400040               43:        j max                   #call the max proc...
     0x00400028 0x8ca80000 lw $8,0x00000000($5)       45: after_max: lw $t0,0($a1)       #load last element...
     0x0040002c 0xac480000 sw $8,0x00000000($2)       47:        sw $t0,0($v0)           #copy last element...
     0x00400030 0xaca30000 sw $3,0x00000000($5)       49:        sw $v1,0($a1)           #copy max value to...
```

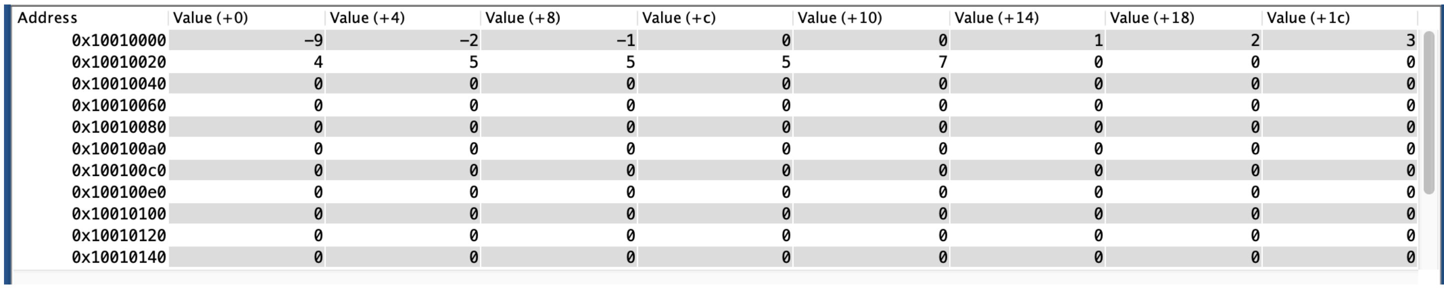
| Label | Address ▲ |
|---|---|
| mips1.asm | |
| main | 0x00400000 |
| after_sort | 0x00400018 |
| end_main | 0x00400020 |
| sort | 0x00400020 |
| after_max | 0x00400028 |
| done | 0x0040003c |
| max | 0x00400040 |
| loop | 0x0040004c |
| ret | 0x0040006c |
| A | 0x10010000 |
| Aend | 0x10010034 |

✓ Data ✓ Text

| Address | Value (+0) | Value (+4) | Value (+8) | Value (+c) | Value (+10) | Value (+14) | Value (+18) | Value (+1c) |
|---|---|---|---|---|---|---|---|---|
| 0x10010000 | 0xfffffff7 | 0xfffffffe | 0xffffffff | 0x00000000 | 0x00000000 | 0x00000001 | 0x00000002 | 0x00000003 |
| 0x10010020 | 0x00000004 | 0x00000005 | 0x00000005 | 0x00000005 | 0x00000007 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010040 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 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 |
| 0x100100e0 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010100 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010120 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010140 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |

0x10010000 (.data) ✓ Hexadecimal Addresses ✓ Hexadecimal Values ASCII

| Name | Number | Value |
|---|---|---|
| $zero | 0 | 0x00000000 |
| $at | 1 | 0x10010000 |
| $v0 | 2 | 0x0000000a |
| $v1 | 3 | 0xfffffffe |
| $a0 | 4 | 0x10010000 |
| $a1 | 5 | 0x10010000 |
| $a2 | 6 | 0x00000000 |
| $a3 | 7 | 0x00000000 |
| $t0 | 8 | 0xfffffffe |
| $t1 | 9 | 0xfffffffe |
| $t2 | 10 | 0x00000000 |
| $t3 | 11 | 0x00000000 |
| $t4 | 12 | 0x00000000 |
| $t5 | 13 | 0x00000000 |
| $t6 | 14 | 0x00000000 |
| $t7 | 15 | 0x00000000 |
| $s0 | 16 | 0x00000000 |
| $s1 | 17 | 0x00000000 |
| $s2 | 18 | 0x00000000 |
| $s3 | 19 | 0x00000000 |
| $s4 | 20 | 0x00000000 |
| $s5 | 21 | 0x00000000 |
| $s6 | 22 | 0x00000000 |
| $s7 | 23 | 0x00000000 |
| $t8 | 24 | 0x00000000 |
| $t9 | 25 | 0x00000000 |
| $k0 | 26 | 0x00000000 |
| $k1 | 27 | 0x00000000 |
| $gp | 28 | 0x10008000 |
| $sp | 29 | 0x7fffeffc |
| $fp | 30 | 0x00000000 |
| $ra | 31 | 0x00000000 |
| pc | | 0x00400020 |
| hi | | 0x00000000 |
| lo | | 0x00000000 |



| Address | Value (+0) | Value (+4) | Value (+8) | Value (+c) | Value (+10) | Value (+14) | Value (+18) | Value (+1c) |
|---|---|---|---|---|---|---|---|---|
| 0x10010000 | -9 | -2 | -1 | 0 | 0 | 1 | 2 | 3 |
| 0x10010020 | 4 | 5 | 5 | 5 | 7 | 0 | 0 | 0 |
| 0x10010040 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x10010060 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0x10010080 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |
| 0x10010140 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- **Kết quả:**
  - Mảng ban đầu: A = {7, -2, 5, 1, 5, 2, 0, -1, -9, 4, 5, 3, 0}
  - Mảng được sắp xếp theo trình tự tăng dần thành: {-9, -2, -1, 0, 0, 1, 2, 3, 4, 5, 5, 5, 7}
- **Debug từng dòng:**

| Step | $pc | Giá trị thanh ghi thay đổi | Ghi chú |
|---|---|---|---|
| 1 | 0x00400004 | $at = 0x10010000 | |
| 2 | 0x00400008 | $a0 = 0x10010000 | $a0 = địa chỉ đầu mảng A |
| 3 | 0x0040000c | $at = 0x10010000 | |
| 4 | 0x00400010 | $a1 = 0x10010034 | |
| 5 | 0x00400014 | $a1 = 0x10010030 | $a1 = địa chỉ cuối mảng con |
| 6 | 0x00400020 | | Nhảy đến sort |
| 7 | 0x00400024 | | Nếu $a0 = $a1 thì nhảy đến done |
| 8 | 0x00400040 | | Nhảy đến max |
| 9 | 0x00400044 | $v0 = 0x10010000 | Khởi tạo địa chỉ của max |
| 10 | 0x00400048 | $v1 = 0x00000007 | Khởi tạo giá trị của max |
| 11 | 0x0040004c | $t0 = 0x10010000 | Khởi tạo con trỏ đến phần tử kế tiếp |
| 12 | 0x00400050 | | Nếu kế tiếp là cuối ($t0 = $a1) thì nhảy đến ret |

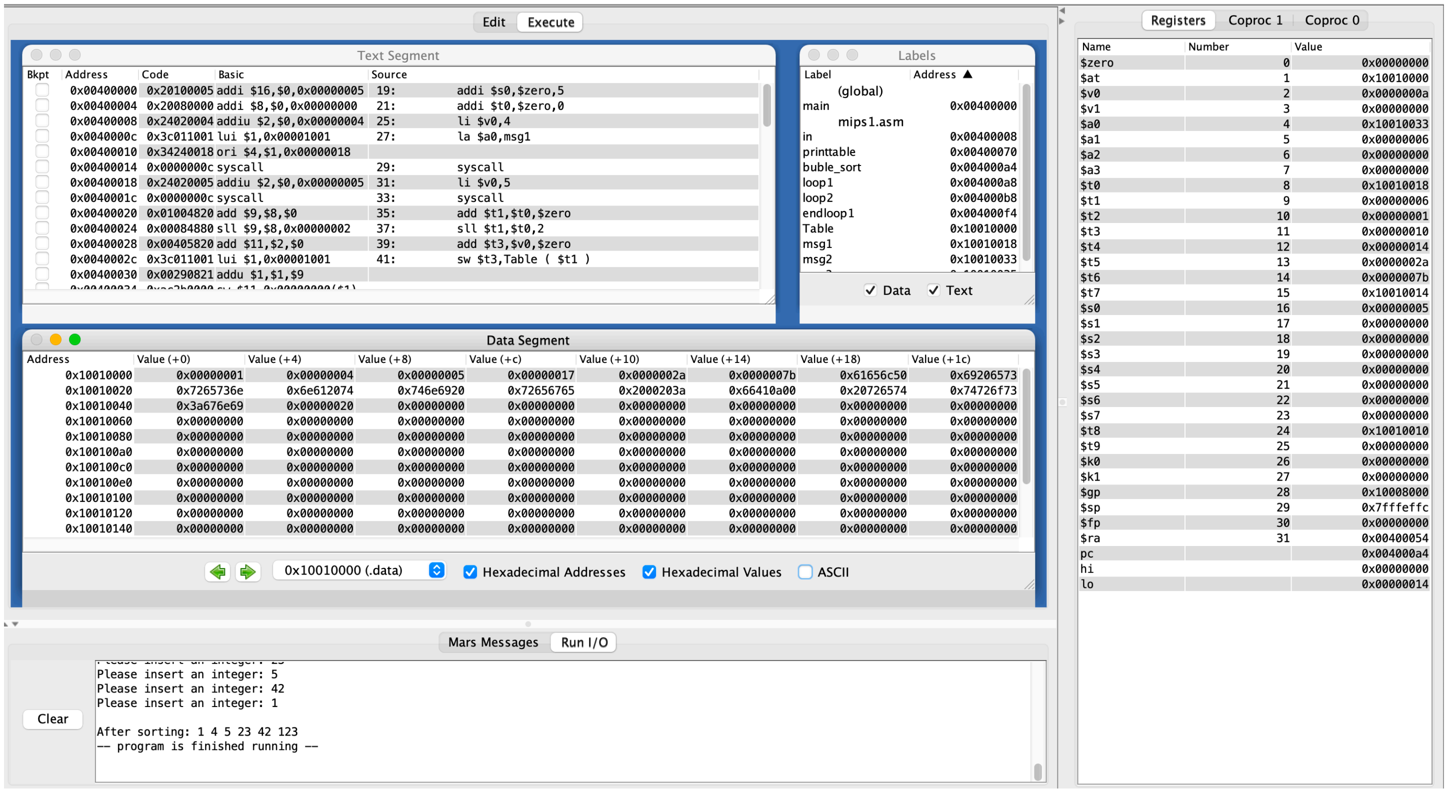
| | | | |
|---|---|---|---|
| 13 | 0x00400054 | $t0 = 0x10010004 | xét phần tử kế tiếp bằng cách tăng địa chỉ của con trỏ địa chỉ thêm 4 |
| 14 | 0x00400058 | $t1 = 0xfffffffe | Giá trị của phần tử kế tiếp |
| 15 | 0x0040005c | $t2= 0x00000001 | $t1<$v1 thì $t2 = 1, ngược lại $t2=0 |
| 16 | 0x00400060 | | Next < max thì nhảy đến loop |
| 17 | 0x0040004c | | |
| 18 | 0x00400050 | $t0 = 0x10010008 | |
| 19 | 0x00400054 | $t0 = 0x10010004 | |
| 20 | 0x00400058 | $t1 = 0x00000005 | |
| 21 | 0x0040005c | $t2 = 0x00000001 | |
| 22 | 0x00400060 | $v0 = 0x10010008 | Địa chỉ max = địa chỉ new-max |
| 23 | 0x00400064 | $t1 = 0x00000005 | Max = new-max |
| 24 | 0x00400068 | | Nhảy đến loop |
| 25 | 0x0040004c | | Bắt đầu quy trình tìm max của mảng con |
| … | … | … | … |

## Assignment 3:

- Chương trình:

```
.data
.align 4
Table: .space 24
msg1: .asciiz "Please insert an integer: "
msg2: .asciiz " "
msg3: .asciiz "\nAfter sorting: "
.text
.globl main
main:
       addi $s0,$zero,5
       addi $t0,$zero,0
in:                   # input
       li $v0,4
       la $a0,msg1
       syscall
       li $v0,5
```

```
        syscall
        add $t1,$t0,$zero
        sll $t1,$t0,2
        add $t3,$v0,$zero
        sw $t3,Table ( $t1 )
        addi $t0,$t0,1
        slt $t1,$s0,$t0
        beq $t1,$zero,in
        la $a0,Table
        addi $a1,$s0,1 #a1=6         #call buble_sort
        jal buble_sort               #print table
        li $v0,4
        la $a0,msg3
        syscall
        la $t0,Table
        #s0=5
        add $t1,$zero,$zero
printtable:                          #print Input
        lw $a0,0($t0)
        li $v0,1
        syscall
        li $v0,4
        la $a0,msg2
        syscall
        addi $t0,$t0,4
        addi $t1,$t1,1
        slt $t2,$s0,$t1
        beq $t2,$zero,printtable
        li $v0,10
        syscall
buble_sort:
        #a0=address of table
        #a1=sizeof table
        add $t0,$zero,$zero #counter1( i )=0
loop1:
        addi $t0,$t0,1               #i++
        bgt $t0,$a1,endloop1         #if t0 < a1 then break;
        add $t1,$a1,$zero               #counter2=size=6
loop2:
        bge $t0,$t1,loop1               #j < = i
        #slt $t3,$t1,$t0
        #bne $t3,$zero,loop1
        addi $t1,$t1,-1                 #j--
        mul $t4,$t1,4                   #t4+a0=table[j]
        addi $t3,$t4,-4                 #t3+a0=table[j-1]
        add $t7,$t4,$a0                 #t7=table[j]
        add $t8,$t3,$a0                 #t8=table[j-1]
        lw $t5,0($t7)
        lw $t6,0($t8)
        bgt $t5,$t6,loop2               #đảo vị trí t5,t6
        sw $t5,0($t8)
        sw $t6,0($t7)
        j loop2
endloop1:
jr $ra
```



Text Segment

| Bkpt | Address | Code | Basic | Source |
|---|---|---|---|---|
| | 0x00400000 | 0x20100005 | addi $16,$0,0x00000005 | 19: addi $s0,$zero,5 |
| | 0x00400004 | 0x20080000 | addi $8,$0,0x00000000 | 21: addi $t0,$zero,0 |
| | 0x00400008 | 0x24020004 | addiu $2,$0,0x00000004 | 25: li $v0,4 |
| | 0x0040000c | 0x3c011001 | lui $1,0x00001001 | 27: la $a0,msg1 |
| | 0x00400010 | 0x34240018 | ori $4,$1,0x00000018 | |
| | 0x00400014 | 0x0000000c | syscall | 29: syscall |
| | 0x00400018 | 0x24020005 | addiu $2,$0,0x00000005 | 31: li $v0,5 |
| | 0x0040001c | 0x0000000c | syscall | 33: syscall |
| | 0x00400020 | 0x01004820 | add $9,$8,$0 | 35: add $t1,$t0,$zero |
| | 0x00400024 | 0x00084880 | sll $9,$8,0x00000002 | 37: sll $t1,$t0,2 |
| | 0x00400028 | 0x00405820 | add $11,$2,$0 | 39: add $t3,$v0,$zero |
| | 0x0040002c | 0x3c011001 | lui $1,0x00001001 | 41: sw $t3,Table ( $t1 ) |
| | 0x00400030 | 0x00290821 | addu $1,$1,$9 | |

Labels

| Label | Address |
|---|---|
| (global) | |
| main | 0x00400000 |
| mips1.asm | |
| in | 0x00400008 |
| printtable | 0x00400070 |
| buble_sort | 0x004000a4 |
| loop1 | 0x004000a8 |
| loop2 | 0x004000b8 |
| endloop1 | 0x004000f4 |
| Table | 0x10010000 |
| msg1 | 0x10010018 |
| msg2 | 0x10010033 |

✓ Data ✓ Text

Data Segment

| Address | Value (+0) | Value (+4) | Value (+8) | Value (+c) | Value (+10) | Value (+14) | Value (+18) | Value (+1c) |
|---|---|---|---|---|---|---|---|---|
| 0x10010000 | 0x00000001 | 0x00000004 | 0x00000005 | 0x00000017 | 0x0000002a | 0x0000007b | 0x61656c50 | 0x69206573 |
| 0x10010020 | 0x7265736e | 0x6e612074 | 0x746e6920 | 0x72656765 | 0x2000203a | 0x66410a00 | 0x20726574 | 0x74726f73 |
| 0x10010040 | 0x3a676e69 | 0x00000020 | 0x00000000 | 0x00000000 | 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 |
| 0x100100e0 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010100 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010120 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |
| 0x10010140 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 | 0x00000000 |

0x10010000 (.data) ✓ Hexadecimal Addresses ✓ Hexadecimal Values ☐ ASCII

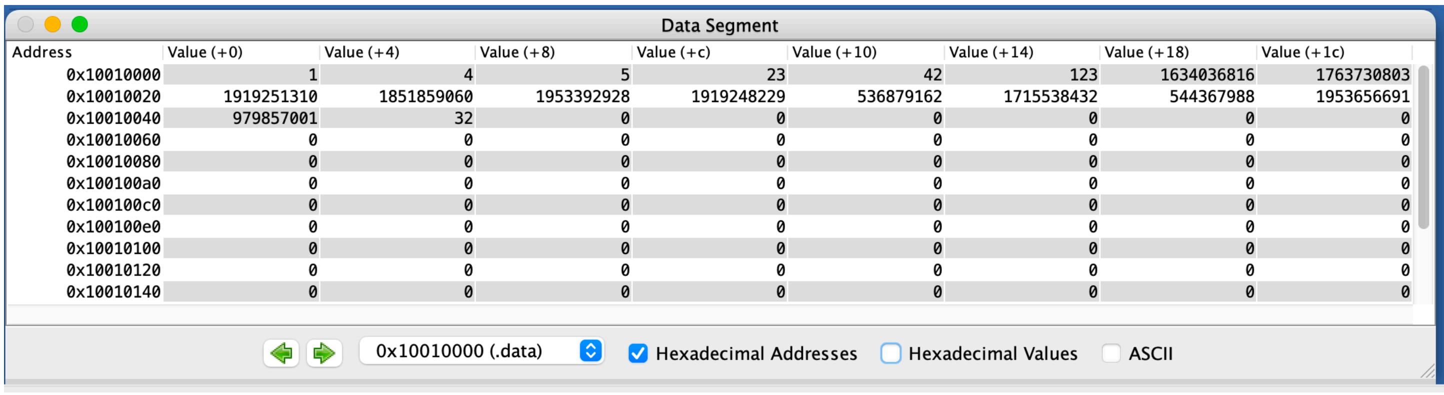
Registers

| Name | Number | Value |
|---|---|---|
| $zero | 0 | 0x00000000 |
| $at | 1 | 0x10010000 |
| $v0 | 2 | 0x0000000a |
| $v1 | 3 | 0x00000000 |
| $a0 | 4 | 0x10010033 |
| $a1 | 5 | 0x00000006 |
| $a2 | 6 | 0x00000000 |
| $a3 | 7 | 0x00000000 |
| $t0 | 8 | 0x10010018 |
| $t1 | 9 | 0x00000006 |
| $t2 | 10 | 0x00000001 |
| $t3 | 11 | 0x00000010 |
| $t4 | 12 | 0x00000014 |
| $t5 | 13 | 0x0000002a |
| $t6 | 14 | 0x0000007b |
| $t7 | 15 | 0x10010014 |
| $s0 | 16 | 0x00000005 |
| $s1 | 17 | 0x00000000 |
| $s2 | 18 | 0x00000000 |
| $s3 | 19 | 0x00000000 |
| $s4 | 20 | 0x00000000 |
| $s5 | 21 | 0x00000000 |
| $s6 | 22 | 0x00000000 |
| $s7 | 23 | 0x00000000 |
| $t8 | 24 | 0x10010010 |
| $t9 | 25 | 0x00000000 |
| $k0 | 26 | 0x00000000 |
| $k1 | 27 | 0x00000000 |
| $gp | 28 | 0x10008000 |
| $sp | 29 | 0x7fffeffc |
| $fp | 30 | 0x00000000 |
| $ra | 31 | 0x00400054 |
| pc | | 0x004000a4 |
| hi | | 0x00000000 |
| lo | | 0x00000014 |

Run I/O

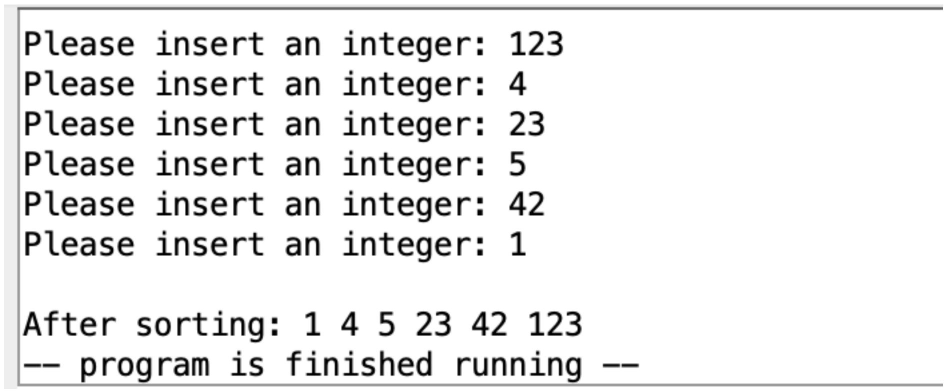
```
Please insert an integer: 5
Please insert an integer: 42
Please insert an integer: 1

After sorting: 1 4 5 23 42 123
-- program is finished running --
```

- Kết quả:
  + Mảng nhập vào là: A = {123, 4, 23, 5, 42, 1}
  + Output của chương trình:



```
Data Segment
Address     Value (+0)  Value (+4)  Value (+8)  Value (+c)  Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000           1           4           5          23          42         123  1634036816  1763730803
0x10010020  1919251310  1851859060  1953392928  1919248229   536879162  1715538432   544367988  1953656691
0x10010040   979857001          32           0           0           0           0           0           0
0x10010060           0           0           0           0           0           0           0           0
0x10010080           0           0           0           0           0           0           0           0
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
0x10010140           0           0           0           0           0           0           0           0
```

0x10010000 (.data) [✓ Hexadecimal Addresses] Hexadecimal Values ASCII



```
Please insert an integer: 123
Please insert an integer: 4
Please insert an integer: 23
Please insert an integer: 5
Please insert an integer: 42
Please insert an integer: 1

After sorting: 1 4 5 23 42 123
-- program is finished running --
```

- **Debug từng dòng:**

| Step | $pc | Giá trị thanh ghi thay đổi |
|---|---|---|
| 1 | 0x00400004 | $s0 = 0x00000005 |
| 2 | 0x00400008 | $t0 = 0x00000000 |
| 3 | 0x0040000c | $v0 = 0x00000004 |
| 4 | 0x00400010 | $at = 0x10010000 |
| 5 | 0x00400014 | $a0 = 0x10010018 |
| 6 | 0x0040001c | $v0 = 0x00000005 |
| 7 | 0x00400028 | $t1 = 0x00000000 |
| 8 | 0x0040002c | $t4 = 0x00000004 |
| 9 | 0x00400030 | $at = 0x10010000 |
| 10 | 0x0040003c | $t0 = 0x00000001 |
| 11 | 0x00400040 | $t1 = 0x00010000 |
| 12 | 0x0040000c | $t0 = 0x00000004 |
| … | … | … |

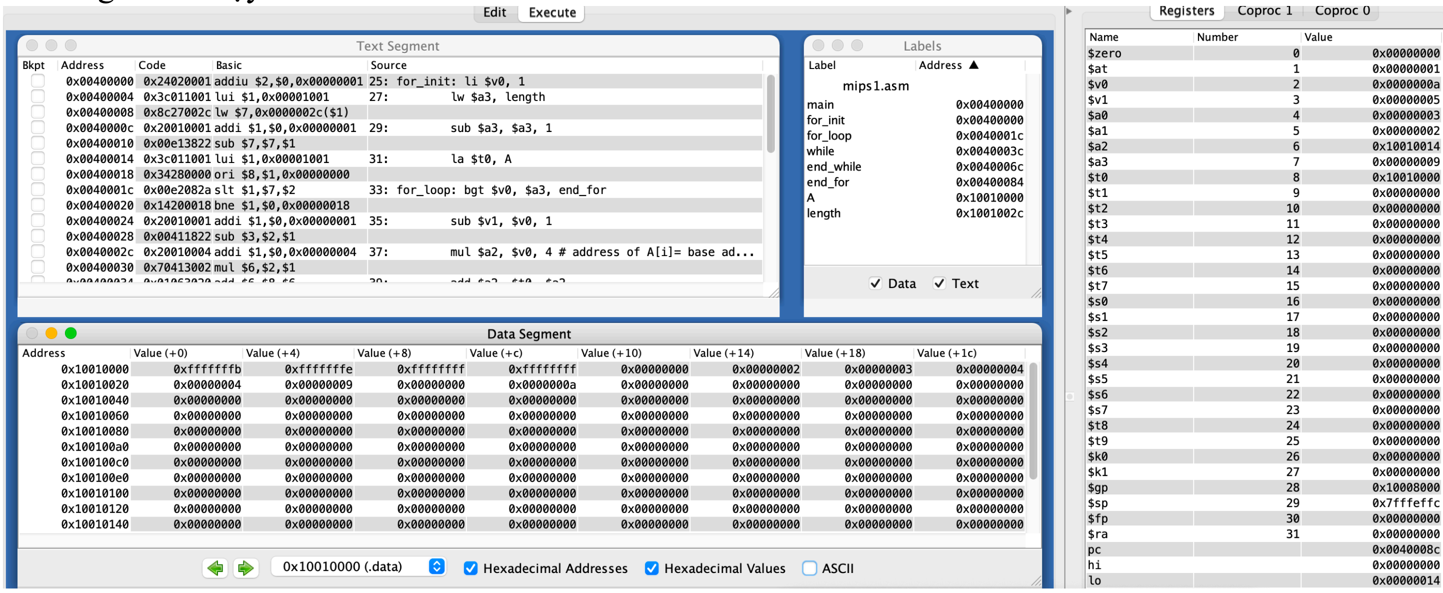
## Assignment 4:

- Chương trình:

```
.data
A: .word -1, -2, 4, 2, 0, -1, 9, 4, -5, 3, 0
length: .word 10
 .text
main:
# Use $v0 to hold firstUnsortedIndex
# Use $v1 to hold testIndex
# Use $a0 to hold elementToInsert
# Use $a1 to hold value of A[ .. ]
# Use $a2 to calculate the address of A[ ... ] in
# Use $a3 to hold the value of (length-1)
# Use $t0 to hold the base/starting address of the A array
for_init: li $v0, 1
       lw $a3, length
       sub $a3, $a3, 1
       la $t0, A
for_loop: bgt $v0, $a3, end_for
       sub $v1, $v0, 1
       mul $a2, $v0, 4 # address of A[i]= base addr of A + i*(element size)
       add $a2, $t0, $a2
       lw $a0, 0($a2)
while: blt $v1, 0, end_while
       mul $a2, $v1, 4 # address of A[i]= base addr of A + i*(element size)
       add $a2, $t0, $a2
       lw $a1, 0($a2)
       ble $a1, $a0, end_while
       sw $a1, 4($a2)
       sub $v1, $v1, 1
       j while
end_while:
       mul $a2, $v1, 4 # address of numbers[i]= base addr of numbers + i*(element size)
       add $a2, $t0, $a2
       sw $a0, 4($a2)
       addi $v0, $v0, 1
       j for_loop
end_for:
       li $v0, 10 # system call to exit
```

```
        syscall
```

- Chương trình chạy:



Text Segment

```
Bkpt  Address     Code       Basic                        Source
      0x00400000  0x24020001 addiu $2,$0,0x00000001  25: for_init: li $v0, 1
      0x00400004  0x3c011001 lui $1,0x00001001       27:         lw $a3, length
      0x00400008  0x8c27002c lw $7,0x0000002c($1)
      0x0040000c  0x20010001 addi $1,$0,0x00000001   29:         sub $a3, $a3, 1
      0x00400010  0x00e13822 sub $7,$7,$1
      0x00400014  0x3c011001 lui $1,0x00001001       31:         la $t0, A
      0x00400018  0x34280000 ori $8,$1,0x00000000
      0x0040001c  0x00e2082a slt $1,$7,$2            33: for_loop: bgt $v0, $a3, end_for
      0x00400020  0x14200018 bne $1,$0,0x00000018
      0x00400024  0x20010001 addi $1,$0,0x00000001   35:         sub $v1, $v0, 1
      0x00400028  0x00411822 sub $3,$2,$1
      0x0040002c  0x20010004 addi $1,$0,0x00000004   37:         mul $a2, $v0, 4 # address of A[i]= base ad...
      0x00400030  0x70413002 mul $6,$2,$1
```

Labels

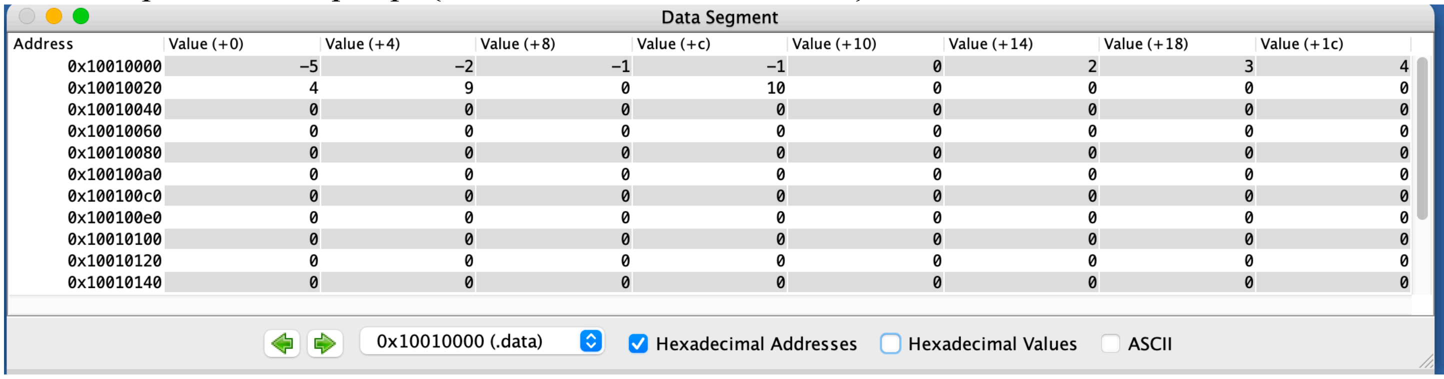
```
Label        Address ▲
     mips1.asm
main         0x00400000
for_init     0x00400000
for_loop     0x0040001c
while        0x0040003c
end_while    0x0040006c
end_for      0x00400084
A            0x10010000
length       0x1001002c

✓ Data   ✓ Text
```

Data Segment

```
Address     Value (+0)  Value (+4)  Value (+8)  Value (+c)  Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000  0xfffffffb  0xfffffffe  0xffffffff  0xffffffff  0x00000000  0x00000002  0x00000003  0x00000004
0x10010020  0x00000004  0x00000009  0x00000000  0x0000000a  0x00000000  0x00000000  0x00000000  0x00000000
0x10010040  0x00000000  0x00000000  0x00000000  0x00000000  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
0x100100e0  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000
0x10010100  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000
0x10010120  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000
0x10010140  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000  0x00000000

0x10010000 (.data)   ✓ Hexadecimal Addresses   ✓ Hexadecimal Values   ☐ ASCII
```

```
Name    Number  Value
$zero        0  0x00000000
$at          1  0x00000001
$v0          2  0x0000000a
$v1          3  0x00000005
$a0          4  0x00000003
$a1          5  0x00000002
$a2          6  0x10010014
$a3          7  0x00000009
$t0          8  0x10010000
$t1          9  0x00000000
$t2         10  0x00000000
$t3         11  0x00000000
$t4         12  0x00000000
$t5         13  0x00000000
$t6         14  0x00000000
$t7         15  0x00000000
$s0         16  0x00000000
$s1         17  0x00000000
$s2         18  0x00000000
$s3         19  0x00000000
$s4         20  0x00000000
$s5         21  0x00000000
$s6         22  0x00000000
$s7         23  0x00000000
$t8         24  0x00000000
$t9         25  0x00000000
$k0         26  0x00000000
$k1         27  0x00000000
$gp         28  0x10008000
$sp         29  0x7fffeffc
$fp         30  0x00000000
$ra         31  0x00000000
pc              0x0040008c
hi              0x00000000
lo              0x00000014
```

- Kết quả:
  - Mảng đầu vào là A = { -1, -2, 4, 2, 0, -1, 9, 4, -5, 3, 0}
  - Ouput sau khi sắp xếp: {-5, -2, -1, -1, 0, 0, 2, 3, 4, 4, 9}



Data Segment

```
Address     Value (+0)  Value (+4)  Value (+8)  Value (+c)  Value (+10) Value (+14) Value (+18) Value (+1c)
0x10010000          -5          -2          -1          -1           0           2           3           4
0x10010020           4           9           0          10           0           0           0           0
0x10010040           0           0           0           0           0           0           0           0
0x10010060           0           0           0           0           0           0           0           0
0x10010080           0           0           0           0           0           0           0           0
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
0x10010140           0           0           0           0           0           0           0           0

0x10010000 (.data)   ✓ Hexadecimal Addresses   ☐ Hexadecimal Values   ☐ ASCII
```

- **Debug từng dòng**:

| Step | $pc | Giá trị thanh ghi thay đổi |
|---|---|---|
| 1 | 0x00400004 | $v0 = 0x00000001 |
| 2 | 0x00400008 | $at = 0x10010000 |
| 3 | 0x0040000c | $a3 = 0x0000000a |
| 4 | 0x00400010 | $at = 0x00000001 |
| 5 | 0x00400014 | $a3 = 0x00000009 |
| 6 | 0x00400018 | $at = 0x10010000 |
| 7 | 0x0040001c | $t0 = 0x10010000 |
| 8 | 0x00400020 | $at = 0x00000000 |

| | | |
|---|---|---|
| 9 | 0x0040002c | $v1 = 0x00000000 |
| 10 | 0x00400030 | $at = 0x00000004 |
| 11 | 0x00400034 | lo = 0x00000004 |
| 12 | 0x00400038 | $a2 = 0x10010004 |
| 8 | 0x00400030 | $a0 = 0xfffffffe |
| 9 | 0x00400040 | $at = 0x00000000 |
| 10 | 0x0040004c | lo = 0x00000000 |
| 11 | 0x00400054 | $a1 = 0xffffffff |
| … | … | … |