Họ và tên : Phạm Ngọc Bảo Anh

MSSV: 20176682

**BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH TUẦN 6**

**Assignment 1:**

*Code:*

.data

A: .word -2, 6, -1, 3, -2

.text

main: la $a0,A

li $a1,5

j mspfx

nop

continue:

lock: j lock

nop

end\_of\_main:

# @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 $v1 to 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:*



* Max-sum prefix: $v1 = -2 + 6 - 1 + 3 = 6
* Độ dài của prefix: $v0 = 4

**Assignment 2:**

*Code:*

.data

A: .word 7, -2, 5, 1, 5, 6, 7, 3, 6, 8, 8, 59, 5

Aend: .word

.text

main: la $a0,A #$a0 = Address(A[0])

la $a1,Aend

addi $a1,$a1,-4

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

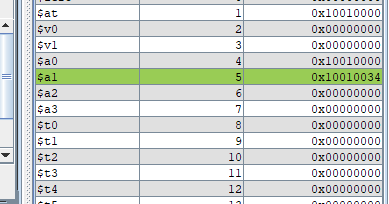
Giải thích :

Trong đó :

$a0 lưu địa chỉ bắt đầu của mảng(địa chỉ A[0])

Aend lưu địa chỉ của phần tử đứng ngay sau phần tử cuối cùng của mảng A

Aend có địa chỉ:



$a1(địa chỉ cuối cùng của phần tử trong mảng ) = 0x10010030 (Aend - 4)

Sau đó tiến hành sắp xếp và thoát khỏi main bằng lệnh exit 10

Nếu $a0 = $a1 => Số phần tử chưa sắp xếp = 1 => Mảng đã sắp xếp xong

Nếu $a0 khác $a1 => tiếp tục đi tìm phần tử lớn nhất => tiến hành đổi chỗ $t0 và $v1

Tiến hành chạy với mảng 8, 5, 9

Ví dụ với vòng lặp đầu tiên trong hàm max:

$v1 = 8

Do vẫn chưa duyệt hết mảng nên vẫn load dữ liệu phần tử tiếp theo vào $t1 => $t1 = 5

Vì $t1 < $v1 nên $t2 = 1 khác 0 nên lại quay về **loop** thực hiện tiếp => tìm ra max = 9

* Lúc này chỉ cần xét mảng gồm 2 phần tử 8, 5 . Trong hàm max tìm được max = 8 sau đó nhảy sang **after\_max** tiến hành đổi chỗ $t0(phần tử cuối cùng) và $v0(phần tử lớn nhất)

*Kết quả:*

**Assignment 3:**

*Code:*

.data

A: .word 7, -2, 5, 1, 5, 6, 7, 3, 6, 8, 8, 59, 5

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:

sort: beq $a0,$a1,done #single element list is sorted

j max #call the max procedure

after\_loop:

addi $a1,$a1,-4 #decrement pointer to last element

j sort #repeat sort for smaller list

done: j after\_sort

max:

addi $t0,$a0,0 #init pointer to first

loop:

beq $t0,$a1,after\_loop #if j = last or n - i -1, return

addi $t1,$t0,0 # init j

addi $t0,$t0,4 # init j + 1

lw $t3,0($t1) #load j element into $t3

lw $t4,0($t0) #load j+1 element into $t4

slt $t5,$t4,$t3 #(A[j+1])<(A[j]) ?

bne $t5,$zero,swap #if A[j+1])<(A[j]), repeat

j loop #if not, repeat

swap:

sw $t3,0($t0)

sw $t4,0($t1)

j loop

Giải thích :

$a0: Lưu địa chỉ của phần tử đầu tiên của mảng

$a1: Lưu địa chỉ của phần tử cuối cùng của mảng chưa sắp xếp

$t0: giá trị tạm thời cho biến j+1, trỏ đến địa chỉ của phần tử sau đang xét

$t1: giá trị tạm thời cho biến j, trỏ đến địa chỉ của phần tử đang xét

$t3: lưu giá trị của phần tử A[j]

$t4: lưu giá trị của phần tử A[j+1]

$t5: lưu kết quả sau khi so sánh A[j] và A[j+1], giá trị = 0 || 1

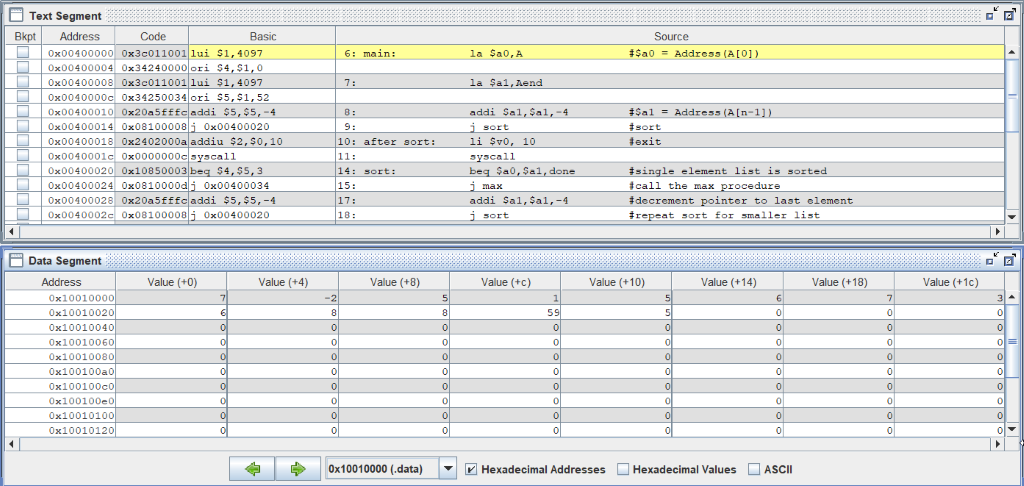
Chạy vòng lặp qua mảng (label #loop), cho đến khi duyệt xong mảng, bỏ qua phần tử cuối cùng của mảng sau đó lại bắt đầu duyệt lại từ đầu mảng còn lại (label #after\_loop).

Trong quá trình duyệt mảng, nếu 2 phần tử cạnh nhau A[j], A[j+1] có thứ tự sai, đổi chỗ (label #swap)

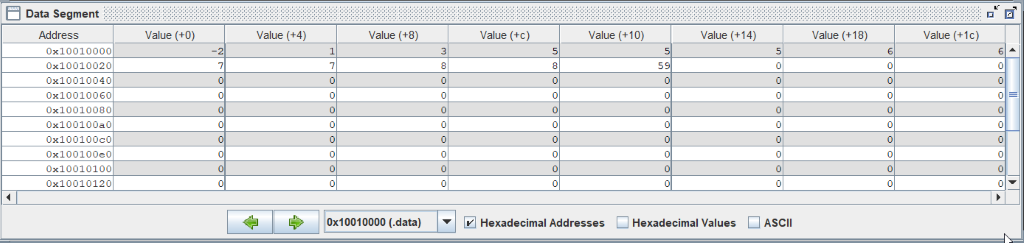
Chương trình dừng lại khi mảng cần sắp xếp chỉ còn lại 1 phần tử (label #done)

*Kết quả:*

+ Mảng khi chưa sắp xếp:



+ Mảng sau khi đã sắp xếp:



**Assignment 4:**

*Code:*

.data

A: .word 7, -2, 5, 1, 5, 6, 7, 3, 6, 8, 8, 59, 5

Aend: .word

.text

main: la $a0,A #$a0 = Address(A[0])

la $a1,Aend # see sort

addi $a2,$a0,4 #$a2 = i = A[1] cuz i run from 1

j sort #sort

after\_sort: li $v0, 10 #exit

syscall

end\_main:

sort: beq $a2,$a1,done # i = n ? or i > n - 1 j loop #call the loop procedure

after\_loop:

addi $a2,$a2,4 #increment pointer of i

j sort #repeat sort

done: j after\_sort

loop:

addi $t0,$a2,-4 #j = i -1

lw $t1,0($a2) # $t1 = key = A[i]

condition1:

slt $t2,$t0,$a0

bne $t2,$0,ret

condition2:

lw $t3,0($t0) #$t3 = A[j]

slt $t2,$t1,$t3 #key < A[j]

beq $t2,$0,ret # if not, ret

sw $t3,4($t0) # A[j+1] = A[j]

addi $t0,$t0,-4 # j = j -1

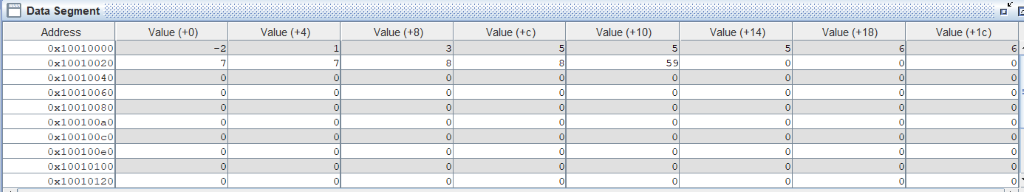
j condition1 #repeat while loop

ret:

sw $t1,4($t0) # A[j+1] = key

j after\_loop

*Kết quả:*



Insertion Sort

* $a0: địa chỉ mảng A
* $a1: địa chỉ sau phần tử cuối mảng A
* $a2: giá trị i
* $t1: địa chỉ A[i]
* $t3: địa chỉ A[j]