Thực hành Kiến trúc máy tính tuần 6

Họ và tên: Nguyễn Khắc Hiếu

MSSV: 20225839

Assignment 1

.data

A: .word 0:100

message1: .asciiz "Nhap so luong phan tu: "

message2: .asciiz "\n"

message3: .asciiz "Tong day con lon nhat: "

.text

main:

la $a0, message1

li $v0, 4

syscall

li $v0,5

syscall

move $a1,$v0 # Nhap cac phan tu cua mang

li $t1,0

la $a0,A

lap:

li $v0,5

syscall

sw $v0,($a0)

addi $t1,$t1,1

addi $a0,$a0,4

blt $t1,$a1,lap

j mspfx

nop

continue:

la $a0, message3

li $v0, 4

syscall

li $v0, 1

move $a0, $v1

syscall

li $v0, 10

syscall

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:

la $a0, A

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

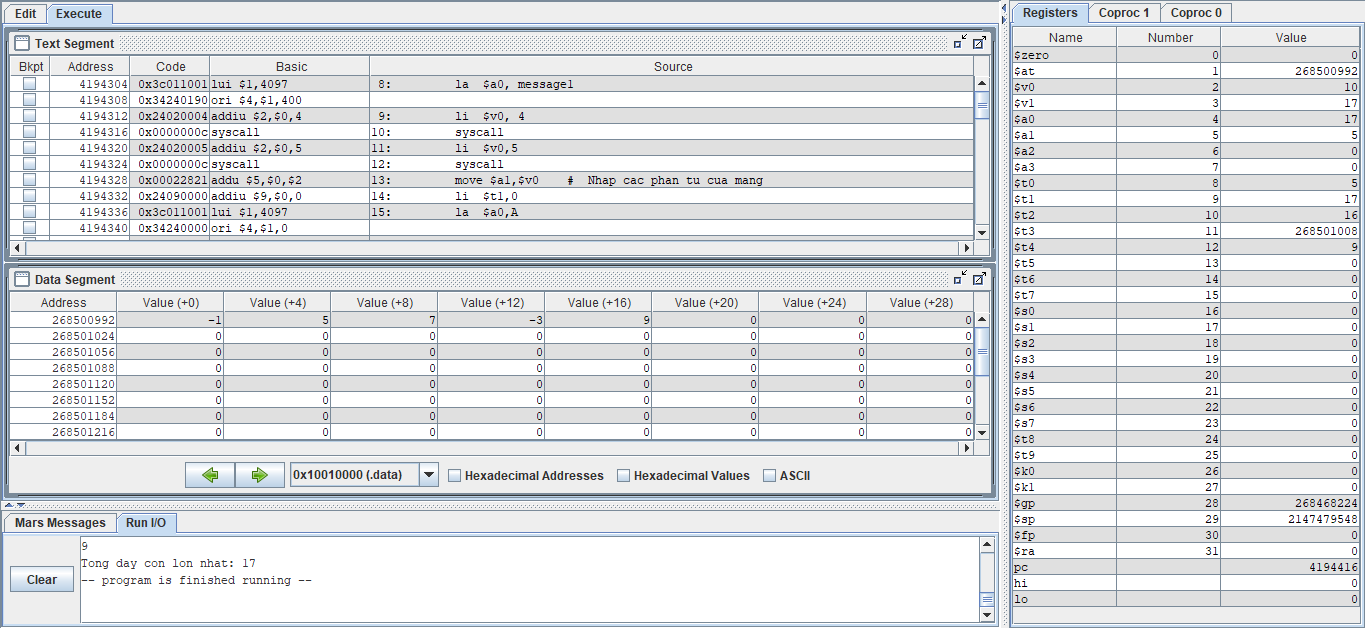
slt $t5,$t0,$a1#set $t5 to 1 if i<n

bne $t5,$zero,loop #repeat if i<n

done:

j continue

mspfx\_end:



* Khi nhập 5 phần tử -1, 5, 7, -3, 9 thì kết quả cho ra là 17 => $v1 = 17
* Kết quả trên đúng với lý thuyết

Assignment 2

.data

A: .word 1,-5,-9,8,6,15,69,-100,77,220

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

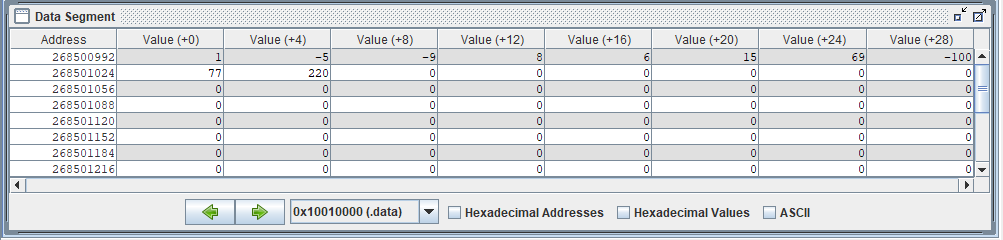
addi $v0,$t0,0 #next element is new max element

addi $v1,$t1,0 #next value is new max value

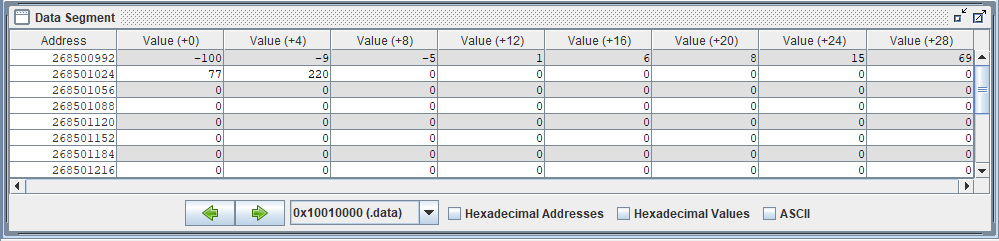
j loop #change completed; now repeat

ret:

j after\_max



* Nhập 10 phần tử: 1,-5,-9,8,6,15,69,-100,77,220
* Mảng sau khi sắp xếp:



* Kết quả đúng với lý thuyết

Assignment 3

.data

A: .word 111,130,231,-428,144,-99,-90,-62,-44,-128

Aend: .word

.text

la $a0, A

la $a1, Aend

li $s0, 0 # count = 0 (count la bien dem phan tu)

li $s1, -1 # i = -1 (i trong loopi)

DemPhanTu:

beq $a1, $a0, Size # So sanh dia chi hien tai trong a1 voi dia chi co so cua mang A

addi $a1, $a1, -4 # dia chi a1 giam de den tung dia chi cua tung phan tu trong mang

addi $s0, $s0, 1 # So luong phan tu tang thêm 1

j DemPhanTu

Size:

addi $t0, $s0, -1 # t0 = So luong phan tu cua mang A - 1

loop1:

addi $s1, $s1, 1 # i++

li $s2, 0 # j = 0 (j trong loop 2)

beq $s1, $t0, Exit # Neu i = size - 1 thì thoat

loop2:

sub $t2, $t0, $s1 # t2 = (size - 1) - i

beq $s2, $t2, loop1 # Neu j = (size - 1) - i thi nhay den loop1

if\_swap:

sll $t3, $s2, 2 # Tính offset cua dia chi A[j]

add $s3, $a0, $t3 # Tính dia chi A[j]

lw $v0, 0($s3) # Load giá tri A[j]

addi $s3, $s3, 4 # Tính dia chi cua A[j+1]

lw $v1, 0($s3) # Load giá tri A[j+1]

sle $t4, $v0, $v1 # Neu A[j] <= A[j+1] thì t4 = 1;

# A[j] > A[j+1] thì t4 = 0

beq $t4, $zero, swap # t4 = 0 thì nhay den swap

addi $s2, $s2, 1

j loop2

swap:

sw $v0, 0($s3) # j++

addi $s3, $s3, -4 # Tính dia chi cua A[j] = dia chi cua A[j+1] - 4

sw $v1, 0($s3) # Ghi A[j] vào A[j+1]

addi $s2, $s2, 1

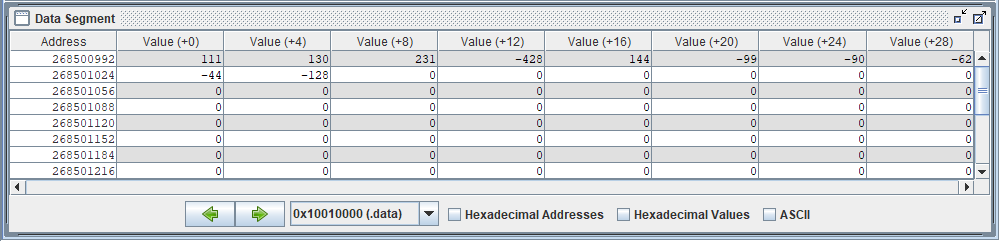
j loop2

Exit:

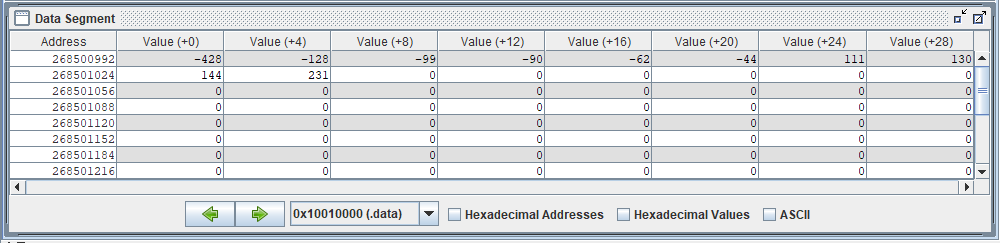
li $v0, 10

syscall

* Nhập 10 phần tử: 1,-5,-9,8,6,15,69,-100,77,220



* Mảng sau khi sắp xếp:



* Kết quả đúng với lý thuyết

Assignment 4

.data

A: .word 47,57,85,137,-51,-205,90,65,425,-482

Aend: .word

.text

la $a0, A

la $a1, Aend

li $s0, 0 # count = 0 (count la bien dem phan tu)

li $s1, 0 # key = 0

li $s2, 0 # j = 0

li $s3, 1 # i = 1

DemPhanTu:

beq $a1, $a0, Loop # So sanh dia chi hien tai trong a1 voi dia chi co so cua mang A

addi $a1, $a1, -4 # Dia chi a1 giam de den tung dia chi cua tung phan tu trong mang

addi $s0, $s0, 1 # So luong phan tu tang thêm 1

j DemPhanTu

Loop:

beq $s3, $s0, Exit # Neu i = So luong phan tu có trong mang thì thoát

sll $t0, $s3, 2 # Tính Offset cua dia chi A[i]

add $s4, $a0, $t0 # Tính dia chi cua A[i]

lw $s1, 0($s4) # Load giá tri A[i] = key

addi $s2, $s3, -1 # j = i - 1

While:

slt $t1, $s2, $zero # Neu j >= 0 thì t1 = 0

sll $t0, $s2, 2 # Tính offset cua dia chi A[j]

add $s5, $a0, $t0 # Tính dia chi cua A[j]

lw $t3, 0($s5) # Load giá tri A[j] = thanh ghi t3

sle $t4, $t3, $s1 # Neu key >= t3 thì t4 = 0

add $t1, $t1, $t4

bne $t1, $zero, loop\_continue # Neu t1 = 0 thì dung while

addi $s5, $s5, 4 # Tính dia chi cua A[j+1]

sw $t3, 0($s5) # Ghi giá tri A[j] vào A[j+1]

addi $s2, $s2, -1 # j = j - 1

j While

loop\_continue:

addi $s5, $s5, 4 # Tính ??a ch? c?a A[j+1]

sw $s1, 0($s5) # Ghi giá tr? key vào A[j+1]

addi $s3, $s3, 1 # i++

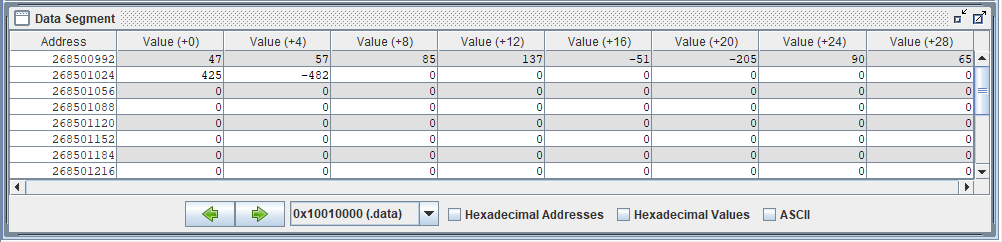
j Loop

Exit:

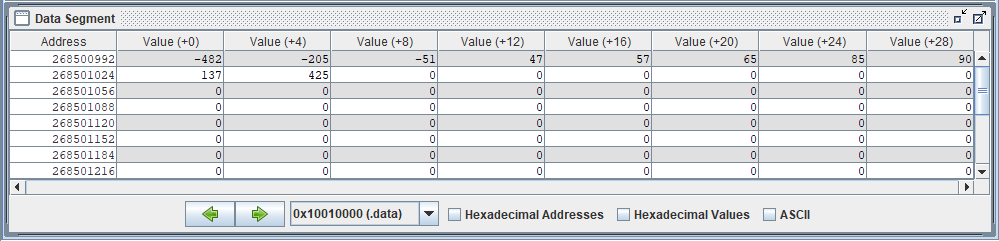
li $v0, 10

syscall

* Nhập 10 phần tử: 47,57,85,137,-51,-205,90,65,425,-482



* Mảng sau khi sắp xếp:



* Kết quả đúng với lý thuyết