Thực hành Kiến trúc máy tính tuần 6

Họ tên: Đỗ Hoàng Minh Hiếu

MSSV: 20225837

Bài 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:

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

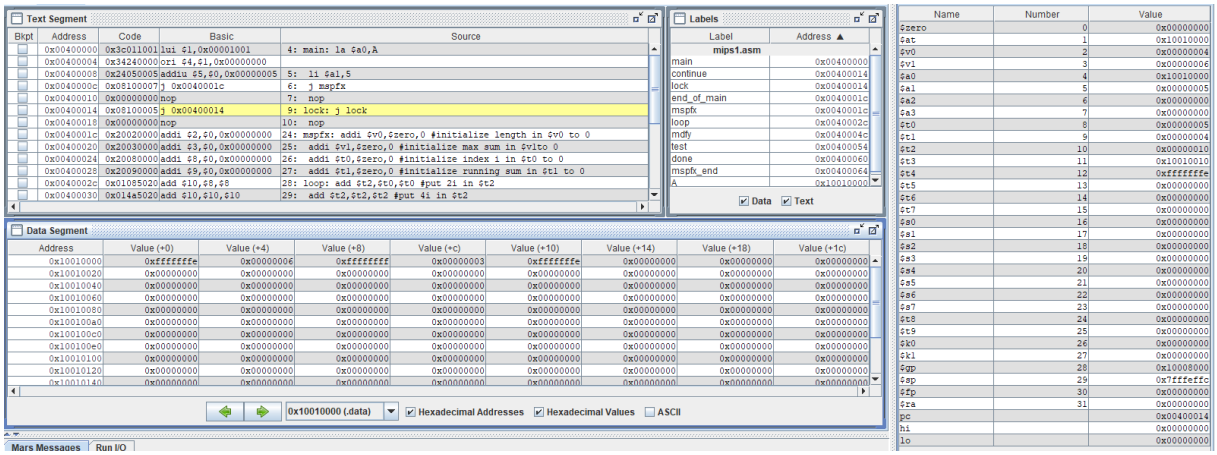
slt $t5,$t0,$a1 #set $t5 to 1 if i<n

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

done: j continue

mspfx_end:

Kết quả sau khi chạy thử:



Sau khi chương trình chạy, ở thanh ghi $v1 cho ra giá trị 6 và $v0 là 4.

Với mảng -2, 6, -1, 3, -2 thì tổng lớn nhất khi cộng 4 phần tử đầu với nhau và bằng 6

=> Chương trình chạy đúng.

Bài 2

.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_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

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
```

Sau khi thực hiện sau mỗi lần lặp loop

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	7	-2	5	1	5	6	7	3
0x10010020	6	8	8	5	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	7	-2	5	1	5	6	7	3
0x10010020	6	8	5	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	7	-2	5	1	5	6	7	3
0x10010020	6	5	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	7	-2	5	1	5	6	5	3
0x10010020	6	7	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	6	-2	5	1	5	6	5	3
0x10010020	7	7	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	6	-2	5	1	5	3	5	6
0x10010020	7	7	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	5	-2	5	1	5	3	6	6
0x10010020	7	7	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	5	-2	5	1	3	5	6	6
0x10010020	7	7	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	5	-2	3	1	5	5	6	6
0x10010020	7	7	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	1	-2	3	5	5	5	6	6
0x10010020	7	7	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	1	-2	3	5	5	5	6	6
0x10010020	7	7	8	8	59	0	0	0

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	-2	1	3	5	5	5	6	6
0x10010020	7	7	8	8	59	0	0	0

=> Chương trình chạy đúng với lý thuyết

Bài 3

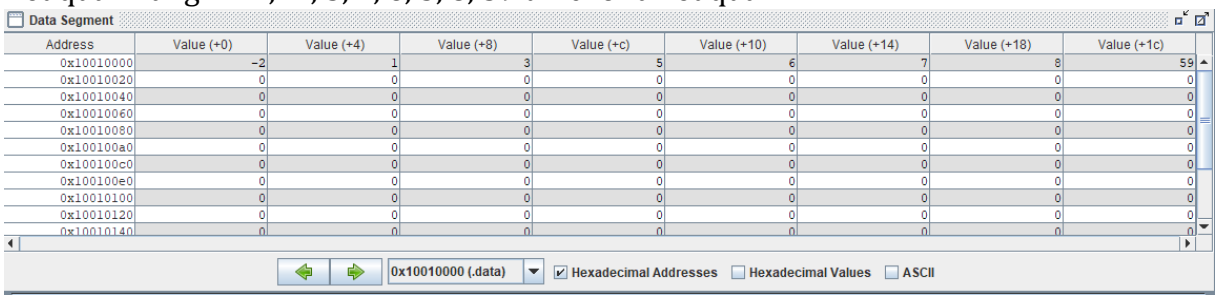
Code:

```
.data
A: .word 7, -2, 5, 1, 6, 3, 8, 59
Aend: .word
.text
main:
   la $a0, A          # $a0 = Address(A[0])
   la $a1, Aend
   addi $a1, $a1, -4   # $a1 = Address(A[n-1])
   li $s0, 1          # Initialize counter to 0
   li $s2, 0
count:
   beq $a1, $a0, endcount  # If $a1 == $a0, end counting
   addi $a1, $a1, -4
   addi $s0, $s0, 1        # Increment counter
   j count
endcount:
   la $a1, Aend
   j bubblesort
after_sort: li $v0, 10 #exit
   syscall

bubblesort:
   addi $s0, $s0, -1   # n--
```

```
    slti $t0, $s0, 2     # if n = 2 -> end sort
    bne $t0, $zero, after_sort
    li $s2, 0   # gan lai $s2 = 0 cho loop1
loop1:
    slt $t1, $s2, $s0
    beq $t1, $zero, bubblesort
    sll $t3, $s2, 2
    add $s3, $a0, $t3 # address A[j]
    lw $v0, 0($s3) # Load A[j]
    addi $s3, $s3, 4 # address A[j+1]
    lw $v1, 0($s3) # Load A[j+1]
    sle $t4, $v0, $v1 # if A[j] <= A[j+1] -> t4 = 1 else t4 = 0
    beq $t4, $zero, swap # t4 = 0 -> swap
    addi $s2, $s2, 1 # j++
    j loop1
swap: sw $v0, 0($s3) # Ghi A[j] vào A[j+1]
    addi $s3, $s3, -4 # address A[j] = address A[j+1] - 4
    sw $v1, 0($s3) # Ghi A[j+1] vào A[j]
    addi $s2, $s2, 1 # j++
    j loop1
```

Kết quả: Mảng A: 7, -2, 5, 1, 6, 3, 8, 59 thì cho ra kết quả



=> Đúng với yêu cầu bài cho

Bài 4

Code:

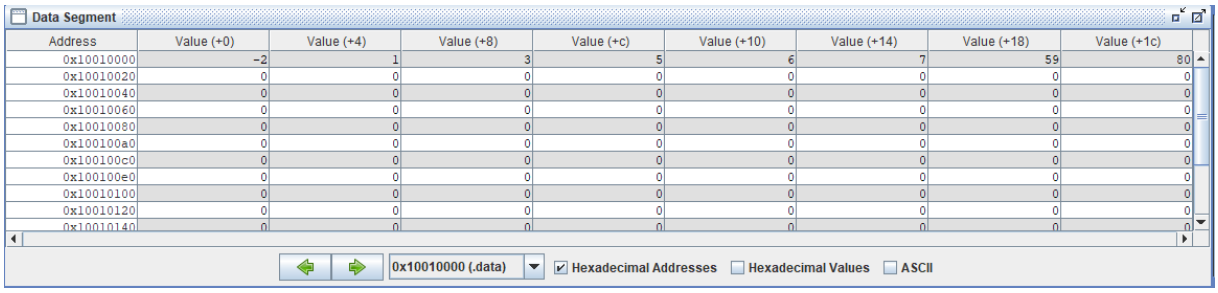
```
.data
A: .word 7, -2, 5, 1, 6, 3, 80, 59
Aend: .word
```

```
.text
main:
   la $a0, A          # $a0 = Address(A[0])
   la $a1, Aend
   addi $a1, $a1, -4   # $a1 = Address(A[n-1])
   li $s0, 1          # Initialize counter to 0
   li $s1, 0      # key = 0
   li $s2, 0      # j = 0
   li $s3, 0      # i = 0
count:
   beq $a1, $a0, endcount  # If $a1 == $a0, end counting
   addi $a1, $a1, -4
   addi $s0, $s0, 1       # Increment counter
   j count
endcount:
   la $a1, Aend
   j insertion_sort
after_sort:
   li $v0, 10 #exit
   syscall

insertion_sort:
   beq $s3, $s0, after_sort # i = n -> aftersort
   sll $t0, $s3, 2
   add $s4, $a0, $t0 # address A[i]
   lw $s1, 0($s4) # Load A[i] = key
   addi $s2, $s3, -1 # j = i - 1
while:
   slt $t1, $s2, $zero # j >= 0 -> t1 = 0
   sll $t0, $s2, 2
   add $s5, $a0, $t0 # address A[j]
```

```
    lw $t3, 0($s5) # $t3 = A[j]
    sle $t4, $t3, $s1 # key >= t3 -> t4 = 0
    add $t1, $t1, $t4
    bne $t1, $zero, loop # t1 = 0 -> stop while (j < 0 && arr[j] <= key)
    addi $s5, $s5, 4 # address A[j+1]
    sw $t3, 0($s5)
    addi $s2, $s2, -1 # j--
    j while
loop:
    addi $s5, $s5, 4 # address A[j+1]
    sw $s1, 0($s5) # A[j+1] = key
    addi $s3, $s3, 1 # i++
    j insertion_sort
```

Kết quả sau khi chạy code:



=> Kết quả đúng.