Báo cáo Tuần 6

Phùng Ngọc Vinh - 20194719

Bài 1:

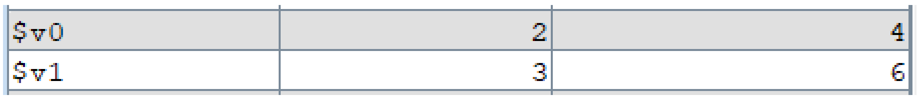
Code:

```
.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:

#---------------------------------------------------------------
#Procedure mspfx
# @brieffind the maximum-sum prefix in a list of integers
# @param[in] a0the base address of this list(A) need to be
processed
# @param[in] a1the number of elements in list(A)
# @param[out] v0the length of sub-array of A in which max
sum reachs.
# @param[out] v1the 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
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:



$v0	2	4
$v1	3	6

Mảng A = {-2,6,-1,3,-2}

Length = $v0 = 4

Khi đó max_sum = $v1 = -2 + 6 +(-1) + (-3) = 6

Bài 2:

* Sắp xếp tăng dần

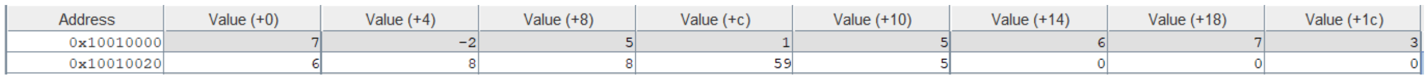
Code:

```
.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:
#-------------------------------------------------------------
```

```
#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
```

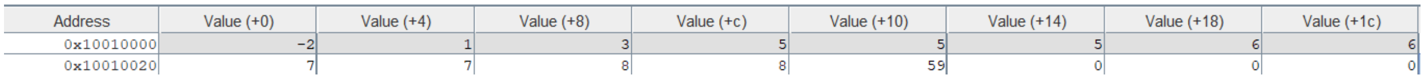
Kết quả:

Trước khi sắp xếp:



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	59	5	0	0	0

Sau khi sắp xếp:



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

* Sắp xếp giảm dần

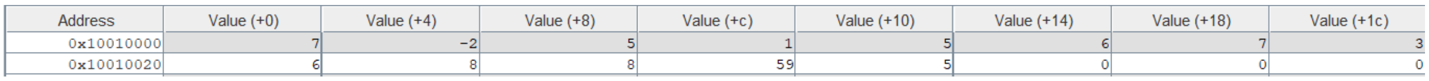
Code:

```
.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 $v0, 10             #exit
   syscall
end_main:
#---------------------------------------------------------------
#procedure sort (ascending selection sort using pointer)
#register usage in sort program
#$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
#$v0 pointer to min element in unsorted part
#$v1 value of min element in unsorted part
#---------------------------------------------------------------
sort:
   beq $a0,$a1,done #single element list is sorted
   j min #call the min procedure
after_min:
   lw $t0,0($a1) #load last element into $t0
   sw $t0,0($v0) #copy last element to min location
   sw $v1,0($a1) #copy min 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 min
#function: fax the value and address of min element in the list
#$a0 pointer to first element
#$a1 pointer to last element
#---------------------------------------------------------------------
min:
   addi $v0,$a0,0     #init min pointer to first element
   lw $v1,0($v0)      #init min 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 $t2,$v1,$t1    #(next)>(min) ?
   bne $t2,$zero,loop      #if (next)>(min), repeat
   addi $v0,$t0,0     #next element is new min element
   addi $v1,$t1,0     #next value is new min value
   j loop          #change completed; now repeat
ret:
   j after_min
```

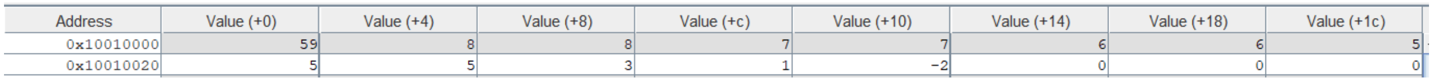
Kết quả:

Trước khi sắp xếp:



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	59	5	0	0	0

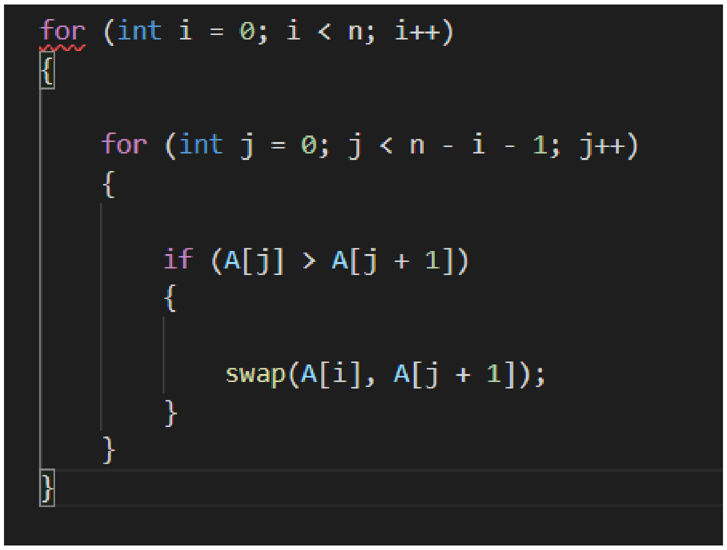
Sau khi sắp xếp:



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

Bài 3:

Mã C:



```c
for (int i = 0; i < n; i++)
{

    for (int j = 0; j < n - i - 1; j++)
    {

        if (A[j] > A[j + 1])
        {

            swap(A[i], A[j + 1]);
        }
    }
}
```

* Sắp xếp tăng dần

Code:

```
.data
A:	.word	7, -2, 5, 1, 5,6,7,3,6,8,8,59,5
.text
main:
	la	$a0, A	# $a0 := dia chi cua A[0]
	li	$s0, 13 # do dai mang n := 13
	j	sort
```

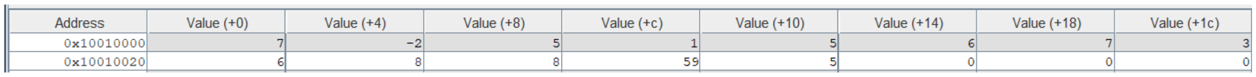
```
complete:
        li      $v0, 10    # exit
        syscall
end_main:


#Thuat toan sap xep noi bot
sort:
        li      $t0, 0       # i = 0
loop1:
        slt     $v0, $t0, $s0
        beq  $v0, $0, end_loop1 # neu i >= n dung loop1
        li      $t1, 0                   # j = 0
loop2:
        sub  $t2, $s0, 1
        sub  $t2, $t2, $t0            # t2 = n-i-1
        slt     $v0, $t1, $t2
        beq  $v0, $0,   end_loop2              #neu j >= n-i-1 dung
loop2
if:
        sll     $t5, $t1, 2               #t5 = 4*j (offset)
        add  $t5, $t5, $a0           #t5 := dia chi cua A[j]
        lw     $t3, 0($t5)              #t3 := A[j]
        lw     $t4, 4($t5)              #t4 := A[j+1]
        sgt   $v0, $t3, $t4
        beq  $v0, $0, end_if              #neu A[j] <= A[j+1] thi end_if
        j       swap
```

```
end_if:
        addi $t1, $t1, 1                # j = j + 1
        j       loop2
end_loop2:
        addi $t0, $t0, 1                # i = i + 1
        j       loop1
end_loop1:
        j       complete
swap:
        sw    $t3, 4($t5)
        sw    $t4, 0($t5)
        j        end_if
```

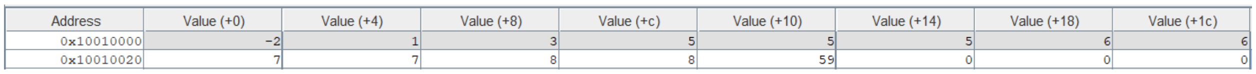
Kết quả:

Trước khi sắp xếp:



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	59	5	0	0	0

Sau khi sắp xếp:



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

* Sắp xếp giảm dần

Code:

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

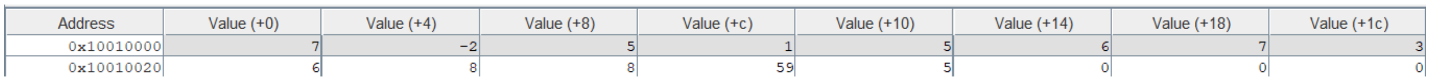
```
main:
        la      $a0, A      # $a0 := dia chi cua A[0]
        li      $s0, 13 # do dai mang n := 13
        j       sort
complete:
        li      $v0, 10     # exit
        syscall
end_main:



#Thuat toan sap xep noi bot
sort:
        li      $t0, 0      # i = 0
loop1:
        slt     $v0, $t0, $s0
        beq     $v0, $0, end_loop1 # neu i >= n dung loop1
        li      $t1, 0             # j = 0
loop2:
        sub     $t2, $s0, 1
        sub     $t2, $t2, $t0      # t2 = n-i-1
        slt     $v0, $t1, $t2
        beq     $v0, $0,    end_loop2           #neu j >= n-i-1 dung
loop2
if:
        sll     $t5, $t1, 2        #t5 = 4*j (offset)
        add     $t5, $t5, $a0      #t5 := dia chi cua A[j]
        lw      $t3, 0($t5)        #t3 := A[j]
        lw    $t4, 4($t5)           #t4 := A[j+1]
        sgt   $v0, $t4, $t3
        beq   $v0, $0, end_if            #neu A[j] >= A[j+1] thi end_if
        j     swap
end_if:
        addi $t1, $t1, 1            # j = j + 1
        j     loop2
end_loop2:
        addi $t0, $t0, 1            # i = i + 1
        j     loop1
end_loop1:
        j     complete
swap:
        sw    $t3, 4($t5)
        sw    $t4, 0($t5)
        j      end_if
```

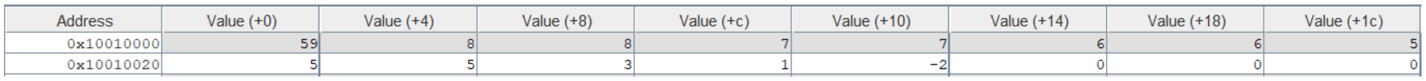
Kết quả:

Trước khi sắp xếp:



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	59	5	0	0	0

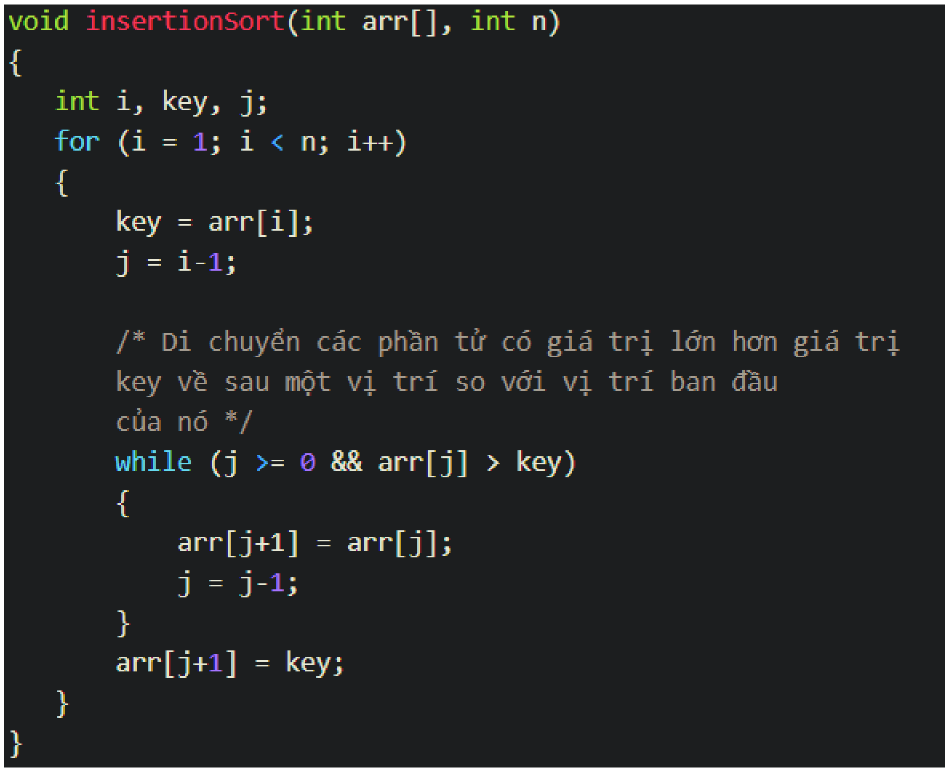
Sau khi sắp xếp:



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

Bài 4:

Mã C:



```c
void insertionSort(int arr[], int n)
{
   int i, key, j;
   for (i = 1; i < n; i++)
   {
       key = arr[i];
       j = i-1;

       /* Di chuyển các phần tử có giá trị lớn hơn giá trị
       key về sau một vị trí so với vị trí ban đầu
       của nó */
       while (j >= 0 && arr[j] > key)
       {
           arr[j+1] = arr[j];
           j = j-1;
       }
       arr[j+1] = key;
   }
}
```

* Sắp xếp tăng dần

Code:

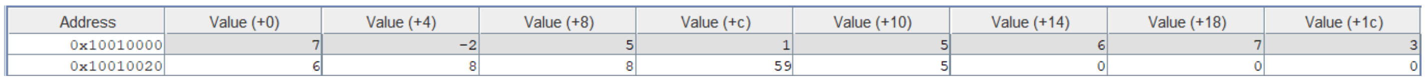
```
.data
A:      .word       7,-2,5,1,5,6,7,3,6,8,8,59,5
.text
        la      $a0, A              # a0 := dia chi A[0]
        li      $s0, 13             # do dai mang n:= 13
        j       sort
complete:
        li      $v0, 10             # exit
        syscall
end_main:

#thuat toan sap xep chen
```

```
sort:
        li      $t0, 1              # i = 1
        li      $t1, 0              # j = 0
        li      $t2, 0              # key = 0
loop:
        slt     $v0, $t0, $s0
        beq     $v0, $0, end_loop   #neu i >= n thi end_loop
        sll     $t3, $t0, 2         # t3 = 4*i
        add     $t3, $t3, $a0       # lay dia chi A[i]
        lw      $t2, 0($t3)         # key = A[i]
        sub     $t1, $t0, 1         # j = i - 1
while:
        bltz    $t1, end_while              #neu j < 0 thi end_while
        sll     $t4, $t1, 2         # t4 = 4*j
        add     $t4, $t4, $a0       # lay di chi A[j]
        lw      $t5, 0($t4)         # t5 = A[j]
        blt     $t5, $t2, end_while# neu A[j] < key thi end_while
        sw      $t5, 4($t4)         # A[j+1] = A[j]
        sub     $t1, $t1, 1         # j = j - 1
        j       while
end_while:
        sll     $t4, $t1, 2         # t4 = 4*j
        add     $t4, $t4, $a0       # t4 := dia chi A[j+1]
        sw      $t2, 4($t4)         # A[j+1] = key
        addi    $t0, $t0, 1         # i = i + 1
        j       loop
end_loop:
	j	complete
```

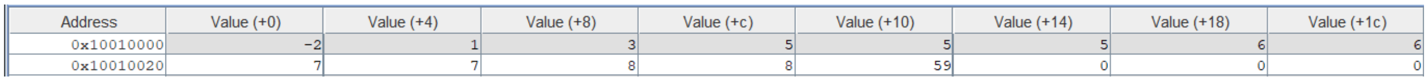
Kết quả:

Trước khi sắp xếp:



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	59	5	0	0	0

Sau khi sắp xếp:



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

* Sắp xếp giảm dần

Code:

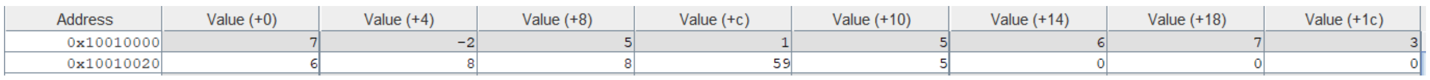
```
.data
A:	.word	7,-2,5,1,5,6,7,3,6,8,8,59,5
.text
	la	$a0, A		# a0 := dia chi A[0]
	li	$s0, 13		# do dai mang n:= 13
	j	sort
complete:
	li	$v0, 10		# exit
	syscall
end_main:

#thuat toan sap xep chen
sort:
	li	$t0, 1		# i = 1
	li	$t1, 0		# j = 0
```

```
        li      $t2, 0              # key = 0
loop:
        slt     $v0, $t0, $s0
        beq     $v0, $0, end_loop   #neu i >= n thi end_loop
        sll     $t3, $t0, 2         # t3 = 4*i
        add     $t3, $t3, $a0       # lay dia chi A[i]
        lw      $t2, 0($t3)         # key = A[i]
        sub     $t1, $t0, 1         # j = i - 1
while:
        bltz    $t1, end_while              #neu j < 0 thi end_while
        sll     $t4, $t1, 2         # t4 = 4*j
        add     $t4, $t4, $a0       # lay di chi A[j]
        lw      $t5, 0($t4)         # t5 = A[j]
        blt     $t2, $t5, end_while # neu A[j] > key thi end_while
        sw      $t5, 4($t4)         # A[j+1] = A[j]
        sub     $t1, $t1, 1         # j = j - 1
        j       while
end_while:
        sll     $t4, $t1, 2         # t4 = 4*j
        add     $t4, $t4, $a0       # t4 := dia chi A[j+1]
        sw      $t2, 4($t4)         # A[j+1] = key
        addi    $t0, $t0, 1         # i = i + 1
        j       loop
end_loop:
        j       complete
```

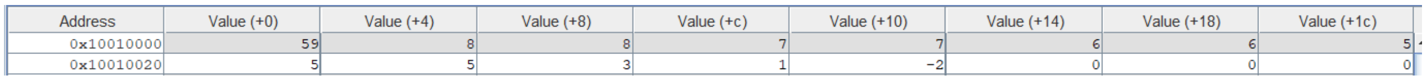
Kết quả:

Trước khi sắp xếp:



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	59	5	0	0	0

Sau khi sắp xếp:



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