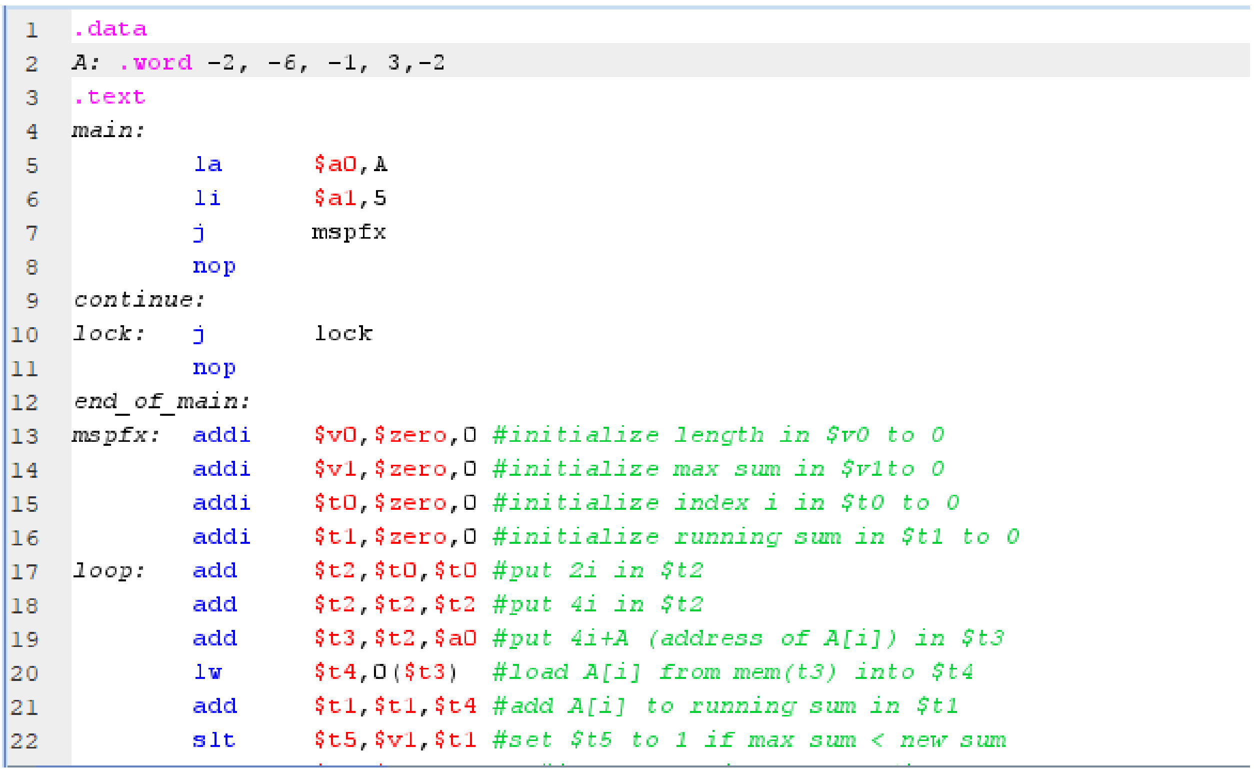
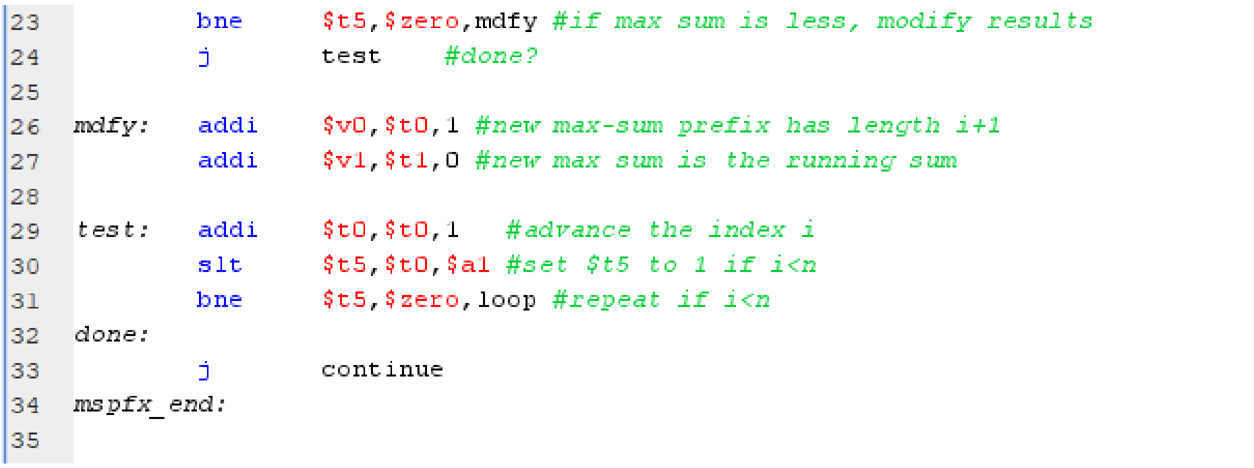
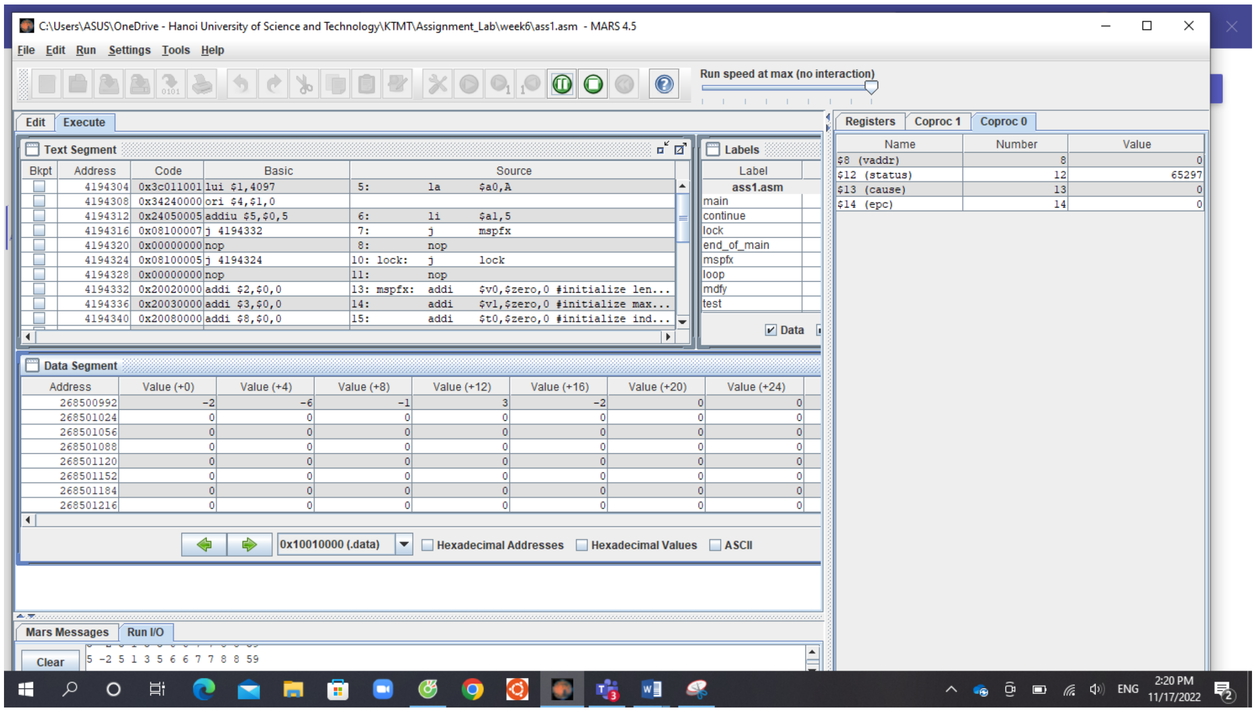
Assignment 1:



```
.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:
```



C:\Users\ASUS\OneDrive - Hanoi University of Science and Technology\KTMT\Assignment_Lab\week6\ass1.asm - MARS 4.5

File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
☐	4194304	0x3c011001	lui $1,4097	5: la $a0,A
☐	4194308	0x34240000	ori $4,$1,0	
☐	4194312	0x24050005	addiu $5,$0,5	6: li $a1,5
☐	4194316	0x08100007	j 4194332	7: j mspfx
☐	4194320	0x00000000	nop	8: nop
☐	4194324	0x08100005	j 4194324	10: lock: j lock
☐	4194328	0x00000000	nop	11: nop
☐	4194332	0x20020000	addi $2,$0,0	13: mspfx: addi $v0,$zero,0 #initialize len...
☐	4194336	0x20030000	addi $3,$0,0	14: addi $v1,$zero,0 #initialize max...
☐	4194340	0x20080000	addi $8,$0,0	15: addi $t0,$zero,0 #initialize ind...

Labels

Label
ass1.asm
main
continue
lock
end_of_main
mspfx
loop
mdfy
test

☑ Data

Registers | Coproc 1 | Coproc 0

Name	Number	Value
$8 (vaddr)	8	0
$12 (status)	12	65297
$13 (cause)	13	0
$14 (epc)	14	0

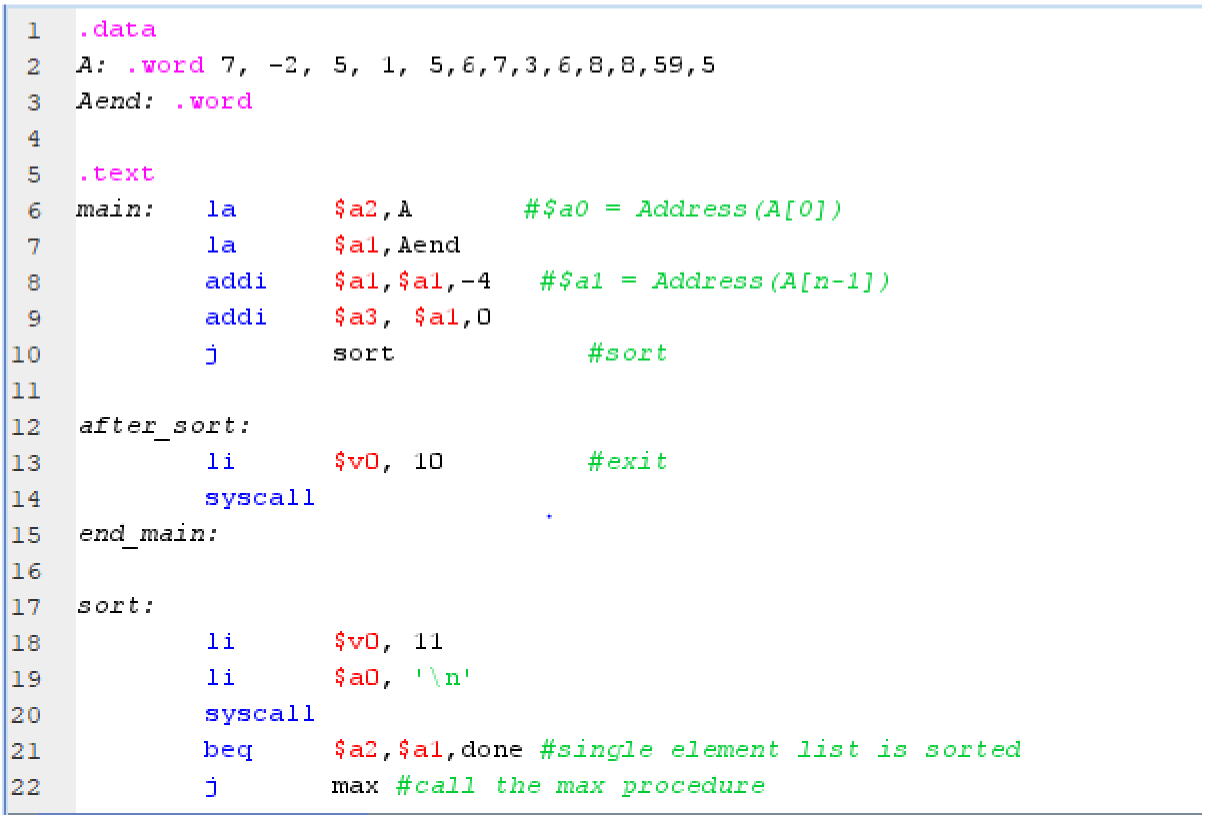
Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)
268500992	-2	-6	-1	3	-2	0	0
268501024	0	0	0	0	0	0	0
268501056	0	0	0	0	0	0	0
268501088	0	0	0	0	0	0	0
268501120	0	0	0	0	0	0	0
268501152	0	0	0	0	0	0	0
268501184	0	0	0	0	0	0	0
268501216	0	0	0	0	0	0	0

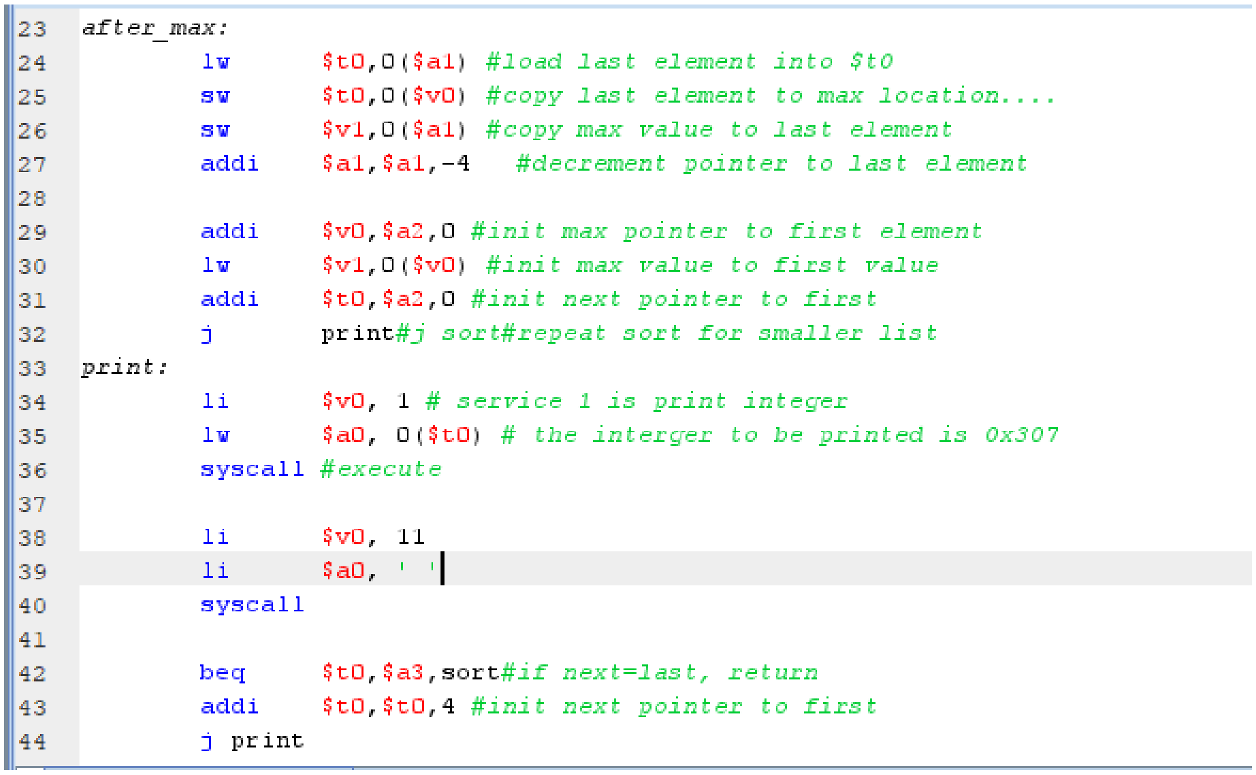
0x10010000 (.data) ☐ Hexadecimal Addresses ☐ Hexadecimal Values ☐ ASCII

```
5 -2 5 1 3 5 6 6 7 7 8 8 59
```

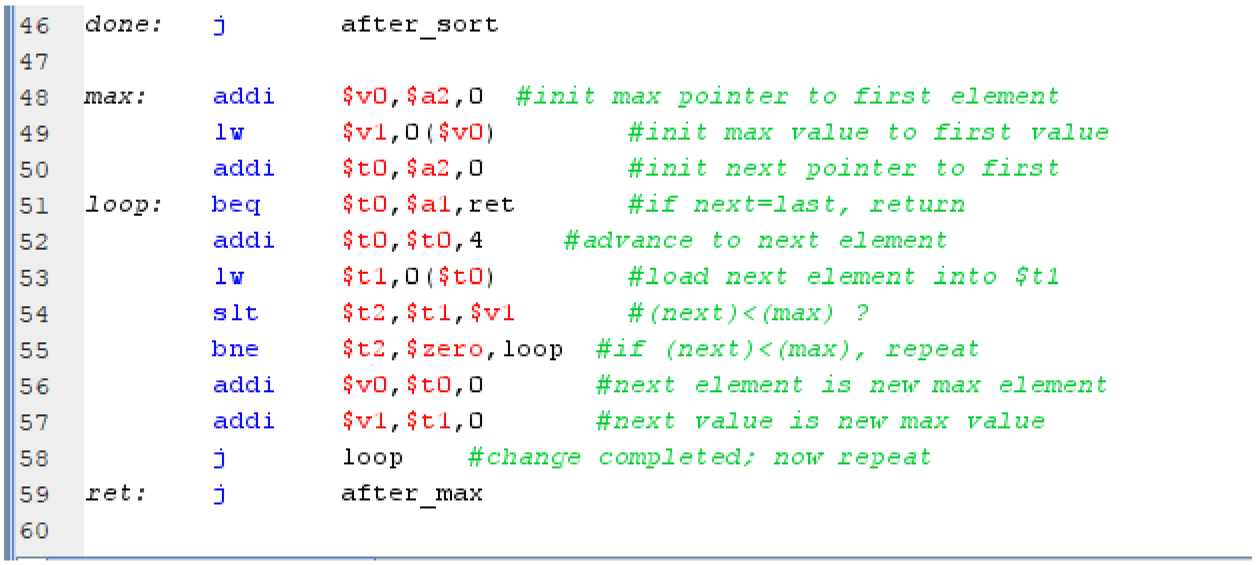
Assignment 2:



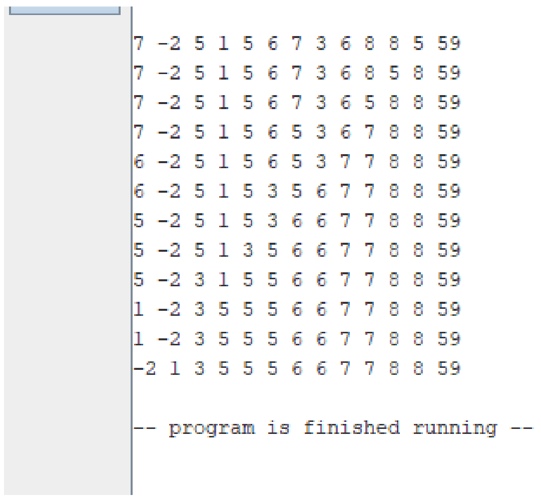
```
.data
A: .word 7, -2, 5, 1, 5,6,7,3,6,8,8,59,5
Aend: .word

.text
main:    la      $a2,A        #$a0 = Address(A[0])
         la      $a1,Aend
         addi    $a1,$a1,-4    #$a1 = Address(A[n-1])
         addi    $a3, $a1,0
         j       sort             #sort

after_sort:
         li      $v0, 10          #exit
         syscall
end_main:

sort:
         li      $v0, 11
         li      $a0, '\n'
         syscall
         beq     $a2,$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

        addi    $v0,$a2,0 #init max pointer to first element
        lw      $v1,0($v0) #init max value to first value
        addi    $t0,$a2,0 #init next pointer to first
        j       print#j sort#repeat sort for smaller list
print:
        li      $v0, 1 # service 1 is print integer
        lw      $a0, 0($t0) # the interger to be printed is 0x307
        syscall #execute

        li      $v0, 11
        li      $a0, ' '
        syscall

        beq     $t0,$a3,sort#if next=last, return
        addi    $t0,$t0,4 #init next pointer to first
        j print
```

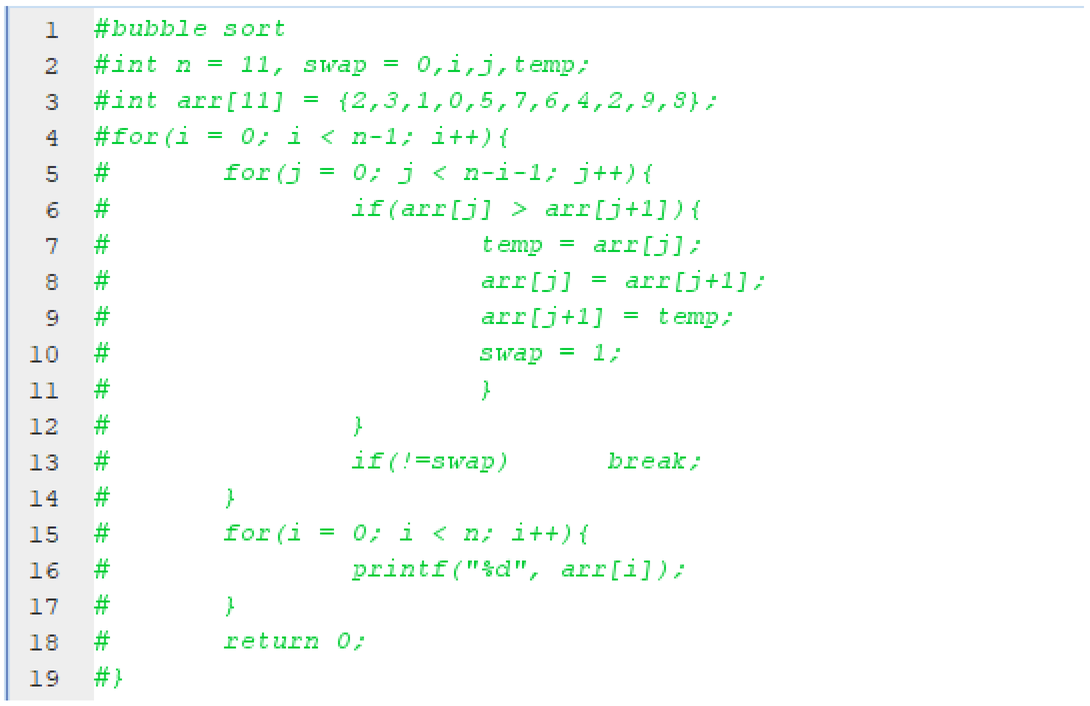


```
done:   j       after_sort

max:    addi    $v0,$a2,0  #init max pointer to first element
        lw      $v1,0($v0)        #init max value to first value
        addi    $t0,$a2,0         #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
```

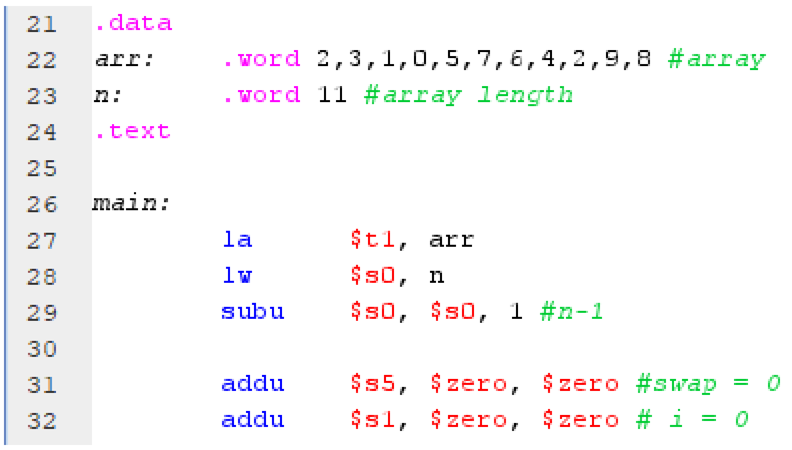


```
7 -2 5 1 5 6 7 3 6 8 8 5 59
7 -2 5 1 5 6 7 3 6 8 5 8 59
7 -2 5 1 5 6 7 3 6 5 8 8 59
7 -2 5 1 5 6 5 3 6 7 8 8 59
6 -2 5 1 5 6 5 3 7 7 8 8 59
6 -2 5 1 5 3 5 6 7 7 8 8 59
5 -2 5 1 5 3 6 6 7 7 8 8 59
5 -2 5 1 3 5 6 6 7 7 8 8 59
5 -2 3 1 5 5 6 6 7 7 8 8 59
1 -2 3 5 5 5 6 6 7 7 8 8 59
1 -2 3 5 5 5 6 6 7 7 8 8 59
-2 1 3 5 5 5 6 6 7 7 8 8 59

-- program is finished running --
```

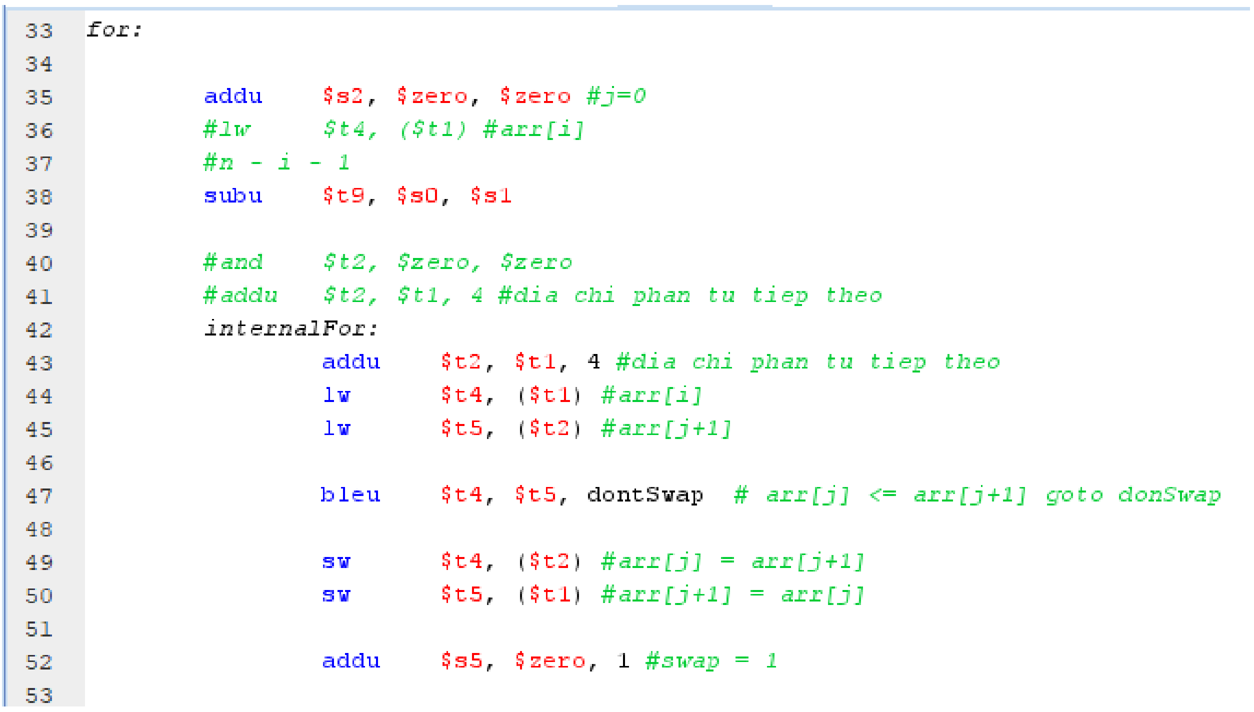
Assignment 3:



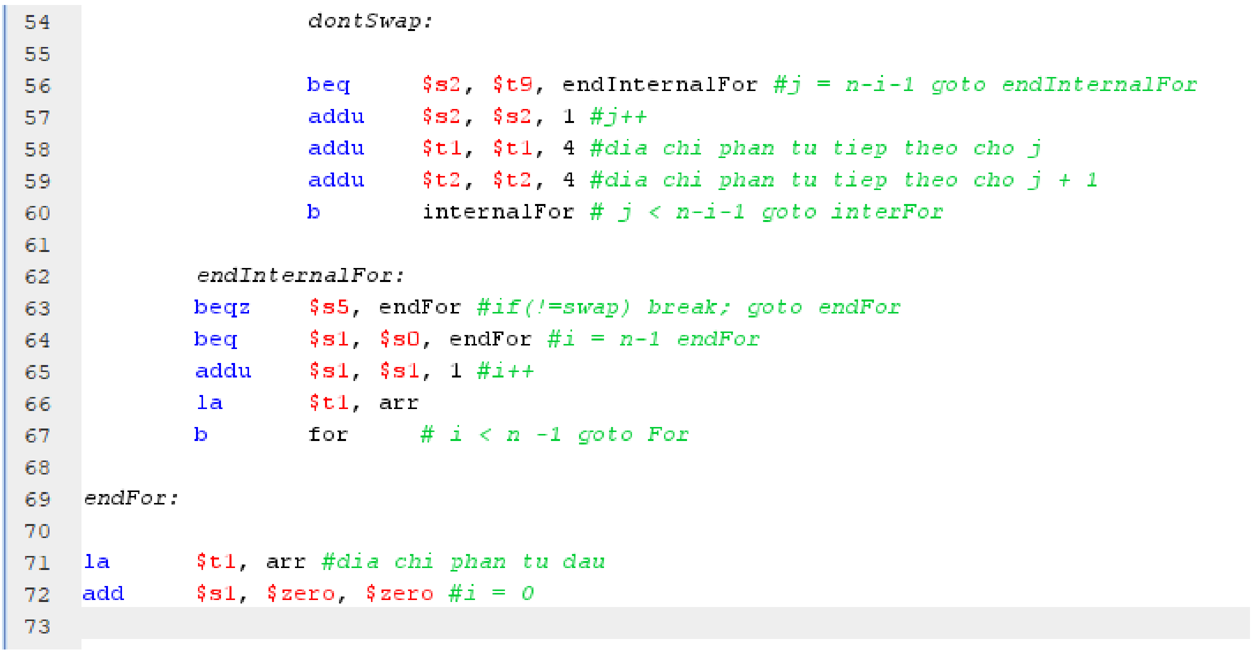
```
#bubble sort
#int n = 11, swap = 0,i,j,temp;
#int arr[11] = {2,3,1,0,5,7,6,4,2,9,8};
#for(i = 0; i < n-1; i++){
#        for(j = 0; j < n-i-1; j++){
#                if(arr[j] > arr[j+1]){
#                        temp = arr[j];
#                        arr[j] = arr[j+1];
#                        arr[j+1] = temp;
#                        swap = 1;
#                        }
#                }
#                if(!=swap)      break;
#        }
#        for(i = 0; i < n; i++){
#                printf("%d", arr[i]);
#        }
#        return 0;
#}
```



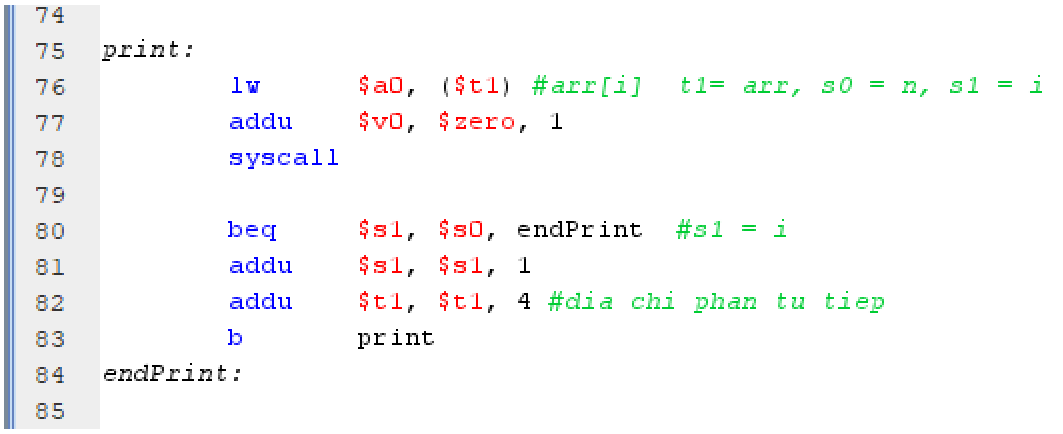
```
.data
arr:    .word 2,3,1,0,5,7,6,4,2,9,8 #array
n:      .word 11 #array length
.text

main:
        la      $t1, arr
        lw      $s0, n
        subu    $s0, $s0, 1 #n-1

        addu    $s5, $zero, $zero #swap = 0
        addu    $s1, $zero, $zero # i = 0
```



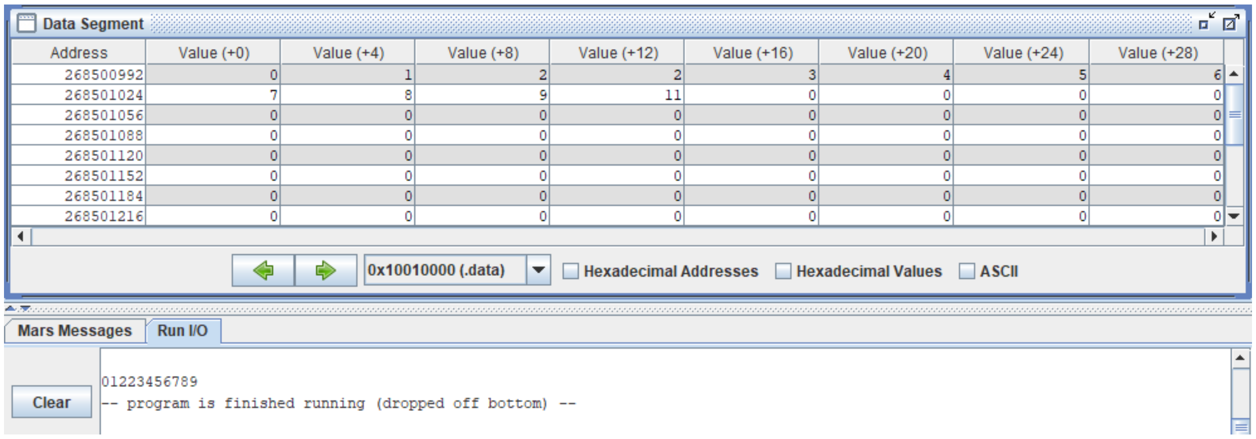
```
for:

        addu    $s2, $zero, $zero #j=0
        #lw     $t4, ($t1) #arr[i]
        #n - i - 1
        subu    $t9, $s0, $s1

        #and    $t2, $zero, $zero
        #addu   $t2, $t1, 4 #dia chi phan tu tiep theo
        internalFor:
                addu    $t2, $t1, 4 #dia chi phan tu tiep theo
                lw      $t4, ($t1) #arr[i]
                lw      $t5, ($t2) #arr[j+1]

                bleu    $t4, $t5, dontSwap  # arr[j] <= arr[j+1] goto donSwap

                sw      $t4, ($t2) #arr[j] = arr[j+1]
                sw      $t5, ($t1) #arr[j+1] = arr[j]

                addu    $s5, $zero, 1 #swap = 1

```



```
                dontSwap:

                beq     $s2, $t9, endInternalFor #j = n-i-1 goto endInternalFor
                addu    $s2, $s2, 1 #j++
                addu    $t1, $t1, 4 #dia chi phan tu tiep theo cho j
                addu    $t2, $t2, 4 #dia chi phan tu tiep theo cho j + 1
                b       internalFor # j < n-i-1 goto interFor

        endInternalFor:
        beqz    $s5, endFor #if(!=swap) break; goto endFor
        beq     $s1, $s0, endFor #i = n-1 endFor
        addu    $s1, $s1, 1 #i++
        la      $t1, arr
        b       for     # i < n -1 goto For

endFor:

la      $t1, arr #dia chi phan tu dau
add     $s1, $zero, $zero #i = 0

```



```
print:
        lw      $a0, ($t1) #arr[i]  t1= arr, s0 = n, s1 = i
        addu    $v0, $zero, 1
        syscall

        beq     $s1, $s0, endPrint  #s1 = i
        addu    $s1, $s1, 1
        addu    $t1, $t1, 4 #dia chi phan tu tiep
        b       print
endPrint:
```

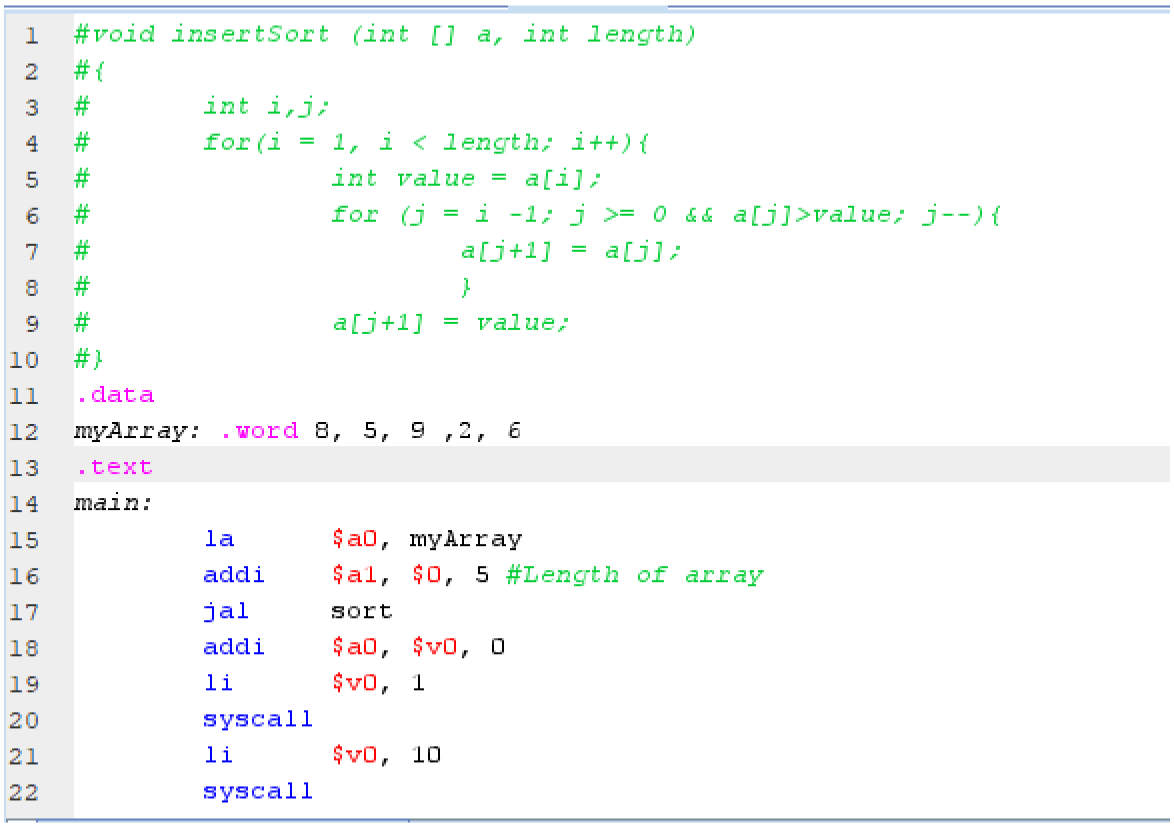


Data Segment

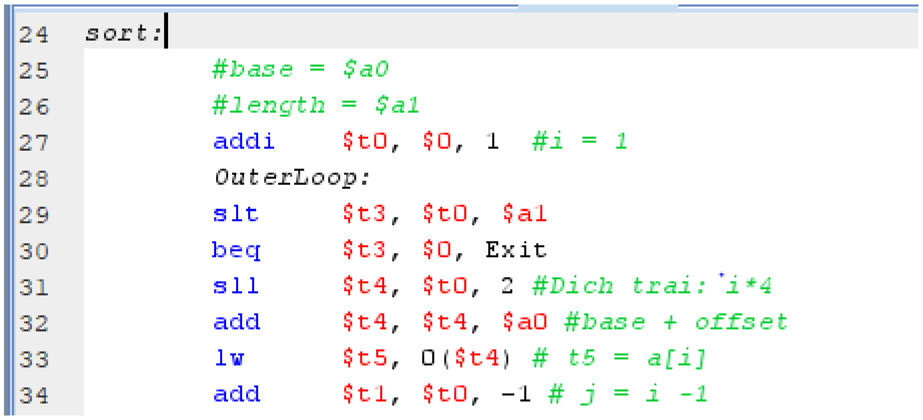
Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
268500992	0	1	2	2	3	4	5	6
268501024	7	8	9	11	0	0	0	0
268501056	0	0	0	0	0	0	0	0
268501088	0	0	0	0	0	0	0	0
268501120	0	0	0	0	0	0	0	0
268501152	0	0	0	0	0	0	0	0
268501184	0	0	0	0	0	0	0	0
268501216	0	0	0	0	0	0	0	0

```
01223456789
-- program is finished running (dropped off bottom) --
```

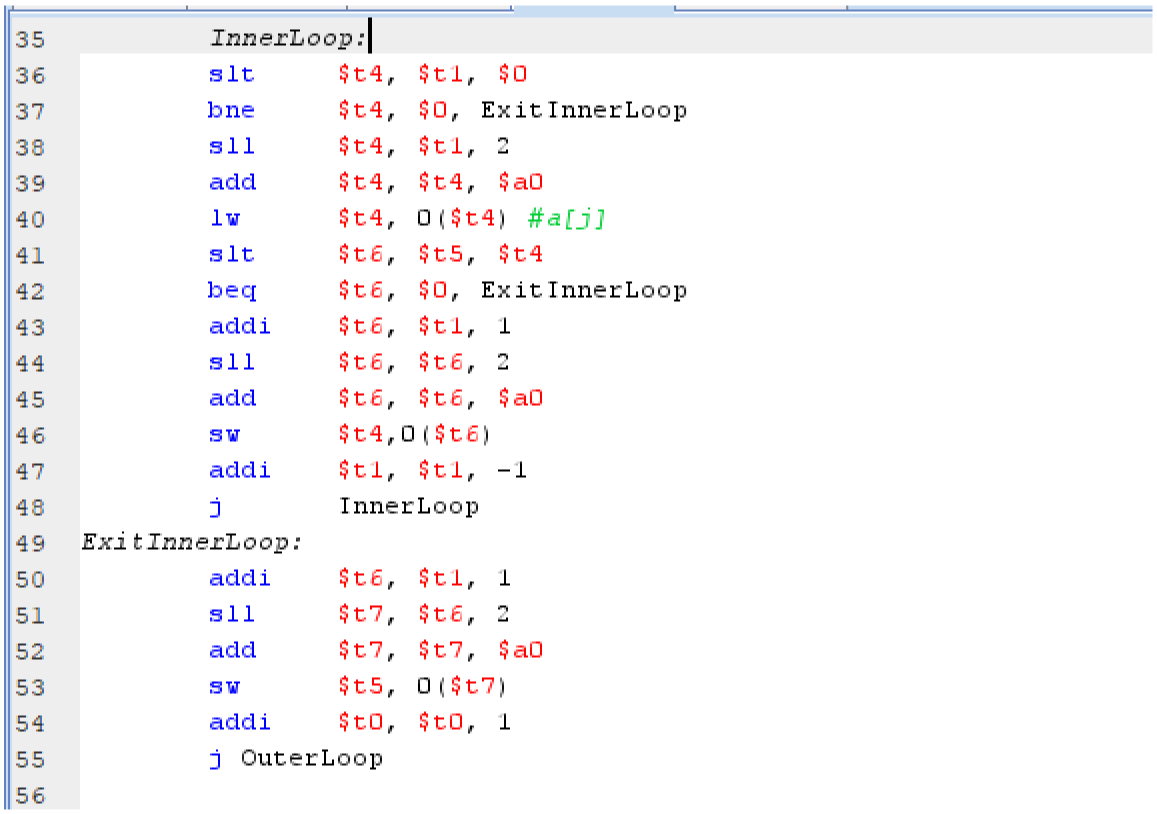
Assignment 4:



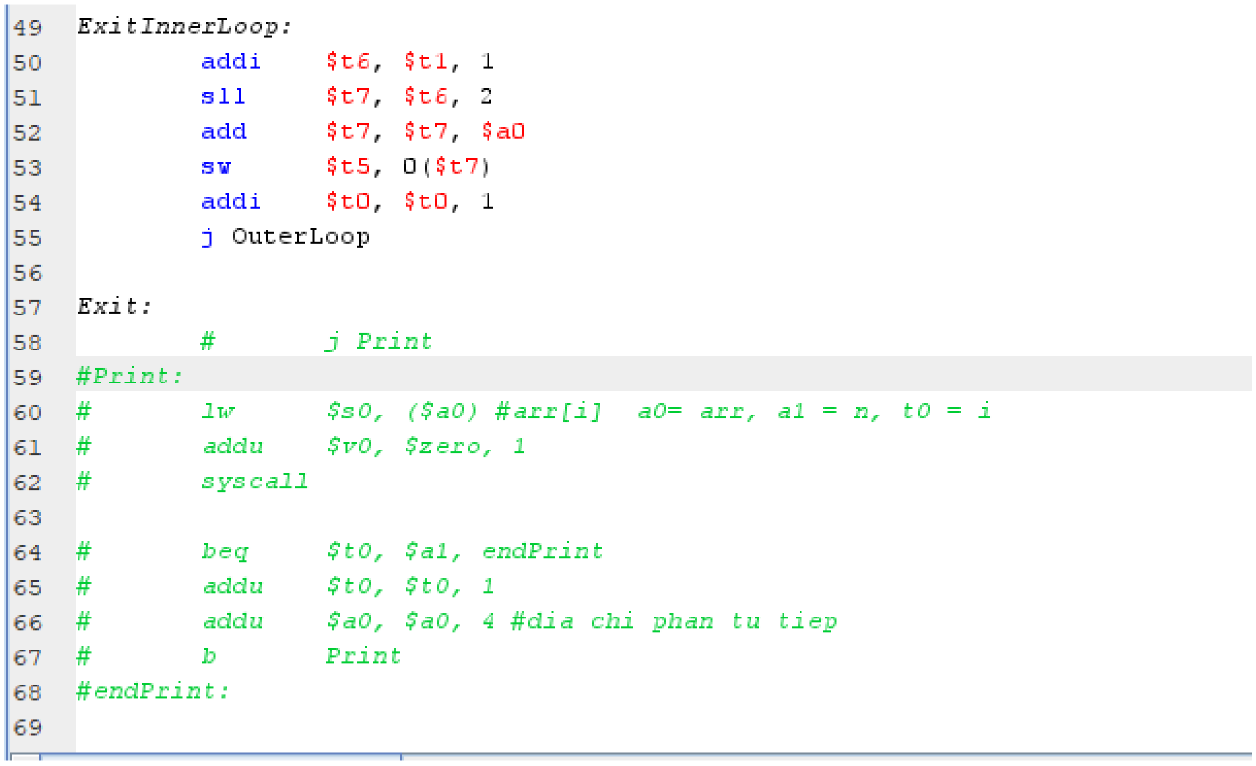
```
#void insertSort (int [] a, int length)
#{
#        int i,j;
#        for(i = 1, i < length; i++){
#                int value = a[i];
#                for (j = i -1; j >= 0 && a[j]>value; j--){
#                        a[j+1] = a[j];
#                        }
#                a[j+1] = value;
#}
.data
myArray: .word 8, 5, 9 ,2, 6
.text
main:
        la      $a0, myArray
        addi    $a1, $0, 5 #Length of array
        jal     sort
        addi    $a0, $v0, 0
        li      $v0, 1
        syscall
        li      $v0, 10
        syscall
```



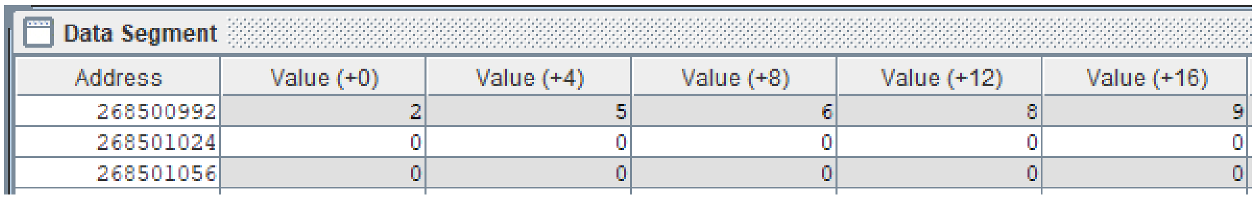
```
sort:
        #base = $a0
        #length = $a1
        addi    $t0, $0, 1  #i = 1
        OuterLoop:
        slt     $t3, $t0, $a1
        beq     $t3, $0, Exit
        sll     $t4, $t0, 2 #Dich trai: i*4
        add     $t4, $t4, $a0 #base + offset
        lw      $t5, 0($t4) # t5 = a[i]
        add     $t1, $t0, -1 # j = i -1
```



```
        InnerLoop:
        slt     $t4, $t1, $0
        bne     $t4, $0, ExitInnerLoop
        sll     $t4, $t1, 2
        add     $t4, $t4, $a0
        lw      $t4, 0($t4) #a[j]
        slt     $t6, $t5, $t4
        beq     $t6, $0, ExitInnerLoop
        addi    $t6, $t1, 1
        sll     $t6, $t6, 2
        add     $t6, $t6, $a0
        sw      $t4,0($t6)
        addi    $t1, $t1, -1
        j       InnerLoop
ExitInnerLoop:
        addi    $t6, $t1, 1
        sll     $t7, $t6, 2
        add     $t7, $t7, $a0
        sw      $t5, 0($t7)
        addi    $t0, $t0, 1
        j OuterLoop

```



```
ExitInnerLoop:
        addi    $t6, $t1, 1
        sll     $t7, $t6, 2
        add     $t7, $t7, $a0
        sw      $t5, 0($t7)
        addi    $t0, $t0, 1
        j OuterLoop

Exit:
        #       j Print
#Print:
#       lw      $s0, ($a0) #arr[i]  a0= arr, a1 = n, t0 = i
#       addu    $v0, $zero, 1
#       syscall

#       beq     $t0, $a1, endPrint
#       addu    $t0, $t0, 1
#       addu    $a0, $a0, 4 #dia chi phan tu tiep
#       b       Print
#endPrint:

```



Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)
268500992	2	5	6	8	9
268501024	0	0	0	0	0
268501056	0	0	0	0	0