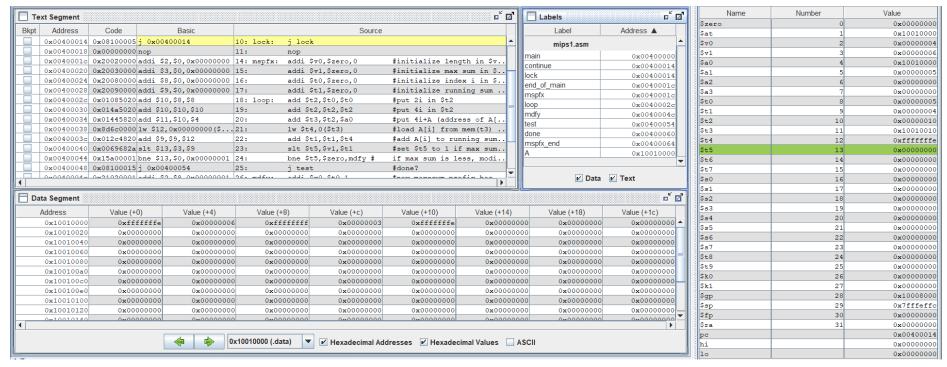
Laboratory Exercise 6

Array and Pointer

1. Assignment 1: Create a new project to implement procedure in Home Assignment 1. Add code the main program and initialize data for the integer list. Compile and upload to simulator. Run this program step by step, observe the process of explore each element of the integer list using indexing method.

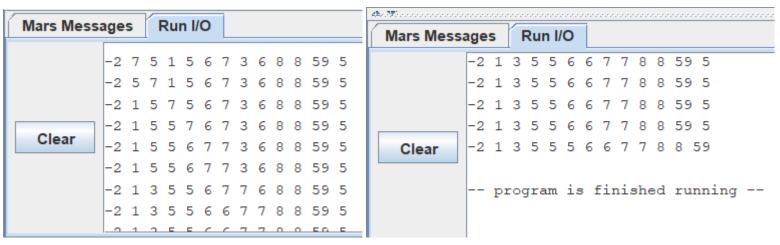
```
Edit
      Execute
 mips1.asm
 1 .data
       .word -2, 6, -1, 3, -2
    .text
 4 main:
            la $a0,A
 5
            li $a1,5
 6
 7
            j mspfx
 8
            nop
   continue:
 9
    lock:
            j lock
11
            nop
    end of main:
13
    mspfx: addi $v0,$zero,0
                                    #initialize length in $v0 to 0
            addi $v1,$zero,0
                                    #initialize max sum in Sv1to 0
15
                                    #initialize index i in $t0 to 0
16
            addi $t0,$zero,0
                                    #initialize running sum in $t1 to 0
17
            addi $t1,$zero,0
            add $t2,$t0,$t0
                                    #put 2i in $t2
            add $t2,$t2,$t2
                                    #put 4i in $t2
19
20
            add $t3,$t2,$a0
                                    #put 4i+A (address of A[i]) in $t3
21
            lw $t4,0($t3)
                                    #load A[i] from mem(t3) into $t4
            add $t1,$t1,$t4
22
                                    #add A[i] to running sum in $t1
23
            slt $t5,$v1,$t1
                                    #set $t5 to 1 if max sum < new sum
24
            bne $t5,$zero,mdfy #
                                    if max sum is less, modify results
            † test
                                    #done?
            addi $v0,$t0,1
                                    #new max-sum prefix has length i+1
    mdfy:
27
            addi $v1,$t1,0
                                    #new max sum is the running sum
    test: addi $t0,$t0,1
                                    #advance the index i
28
            slt $t5,$t0,$a1
                                    #set $t5 to 1 if i<n
29
            bne $t5,$zero,loop
30
                                    #repeat if i<n
           j continue
    done:
32 mspfx end:
```



2. Assignment 2: Create a new project to implement procedure in Home Assignment 2. Add code the main program and initialize data for the integer list. Compile and upload to simulator. Run this program step by step, observe the process of explore each element of the integer list using pointer updating method. Write a procedure to print array after each round.

```
mips1.asm
           mips2.asm
 1 .data
 2 A: .word 7, -2, 5, 1, 5,6,7,3,6,8,8,59,5
 3 Aend: .word
 4 Message: .asciiz "\n"
 5 Messagel: .asciiz " "
   .text
 7 main: la a0, A a0 = Address(A[0])
          la $s5,A #$s5 = Address(A[0])
          la $a1,Aend
 9
10
        addi a1,a1,-4 #$a1 = Address(A[n-1])
          j sort #sort
11
12 after sort:
13
          li $v0, 10 #exit
14
          syscall
15
   end main:
16
17 sort:
18
     slt $s4,$a0,$a1
          beq $a0,$a1,done #single element list is sorted
19
20
         addi $a0,$a0,4
21
        addi $s6,$a0,0
22
          j loop1
   after loop:
23
24
          la $t0,A
25
          la $t1,Aend
26 print:
          lw $at,0($t0)
27
28
          li $v0,1
                        #service 1 is print integer
29
       move $aO,$at
30
        syscall
                        #execute
          li $v0, 4
31
32
          la $aO, Message1
```

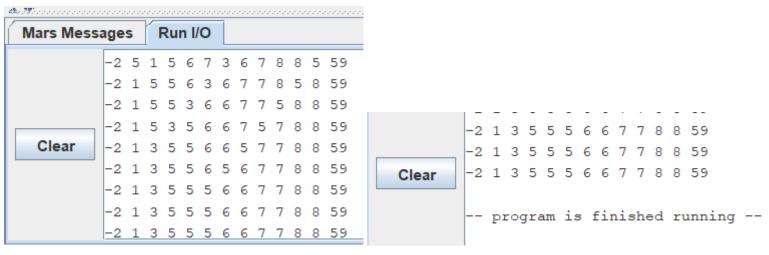
```
33
            syscall
            addi $t0,$t0,4
34
            slt $t3,$t0,$t1
35
            bne $t3,$zero,print
36
37
            li $v0,4
            la $aO, Message
38
39
            syscall
            addi $a0,$s6,0 #$s6 = Address(A[i])
40
41
            sort
            j after sort
42
    done:
    loop1:
43
            addi $v0,$a0,0 #init pointer to current element
44
            lw $v1,0($v0) #init value to current value
45
            addi $t0,$a0,0 #init previous pointer to previous element
46
47
    loop:
            beq $t0,$s5,ret
                                    #if next=last, return
48
            addi $t0,$t0,-4
                                    #advance to previous element
49
                                    #load previous element into $t1
50
            lw $t1,0($t0)
            addi $s0,$v0,0
51
            addi $s1,$v1,0
52
            addi $v0,$t0,0
                                    #previous element is new current element
53
            addi $v1,$t1,0
54
                                    #previous value is new current value
            slt $t2,$t1,$s1
                                    #(previous)<(current) ?
55
            bne $t2,$zero,loop
56
                                    #if (previous) < (current), repeat
57
            lw $s3,0($t0)
58
            sw $s3,0($s0)
            sw $s1,0($t0)
                                    #swap 2 elements
59
            addi $v1,$s1,0
60
            j loop
                                    #change completed; now repeat
61
62
    ret:
63
            i after loop
```



3. Assignment 3: Write a procedure to implement bubble sort algorithm

```
1 .data
 2 A: .word 7, -2, 5, 1, 5,6,7,3,6,8,8,59,5
 3 Aend: .word
 4 Message: .asciiz "\n"
 5 Messagel: .asciiz " "
   . text
 6
 7 main: la $a0,A
                                 #$a0 = Address(A[0])
 8
          la $a1,Aend
          addi $a1,$a1,-4
                             \#$a1 = Address(A[n-1])
 9
           sort
                                 #sort
10
    after sort:
           li $v0, 10
12
                                 #exit
13
           syscall
    end main:
15
16
    sort:
17
           slt $s4,$a0,$a1
           beq $s4,$zero,done
18
                               #single element list is sorted
           j loop1
19
    after loop:
21
           addi $a1,$a1,-4 #decrement pointer to last element
22
           la $t0,A
23
           la $t1,Aend
24 print:
25
           lw $at,0($t0)
26
           li $v0, 1
                                 # service 1 is print integer
           move $a0, $at
27
                                 # execute
28
           syscall
29
           li $v0, 4
30
           la $aO, Message1
31
           syscall
32
           addi $t0,$t0,4
```

```
33
            slt $t3,$t0,$t1
34
           bne $t3,$zero,print
           li $v0, 4
35
36
            la $aO, Message
37
            syscall
                                    #$a0 = Address(A[0])
            la $a0,A
38
            j sort
                                    #repeat sort for smaller list
39
            j after sort
    done:
40
41
    loop1:
42
                                    #init pointer to first element
43
            addi $v0,$a0,0
            lw $v1,0($v0)
                                    #init value to first value
44
            addi $t0,$a0,0
                                    #init next pointer to first
45
   loop:
46
47
            beq $t0,$a1,ret
                                    #if next=last, return
            addi $t0,$t0,4
                                    #advance to next element
48
                                    #load next element into $t1
            lw $t1,0($t0)
49
            addi $s0,$v0,0
50
51
            addi $s1,$v1,0
52
            addi $v0,$t0,0
                                    #next element is new current element
53
            addi $v1,$t1,0
                                    #next value is new current value
            slt $t2,$s1,$t1
                                    #(current) < (next) ?
54
55
            bne $t2,$zero,loop
                                    #if (current) < (next), repeat
            lw $s3,0($t0)
                                    #
56
                                    #
57
            sw $s3,0($s0)
                                    #swap 2 elements
            sw $s1,0($t0)
58
59
            addi $v1,$s1,0
                                    #change completed; now repeat
            j loop
60
61
   ret:
            j after loop
62
```



4. Assignment 4: Write a procedure to implement insertion sort algorithm

```
1 .data
 2 A: .word 7, -2, 5, 1, 5,6,7,3,6,8,8,59,5
 3 Aend: .word
 4 Message: .asciiz "\n"
 5 Messagel: .asciiz " "
   .text
 7 main: la $a0,A
                               #$a0 = Address(A[0])
           la $s5,A
                                 #$s5 = Address(A[0])
 8
           la $a1,Aend
 9
                            \#$a1 = Address(A[n-1])
           addi $a1,$a1,-4
10
           sort
11
                                 #sort
    after sort:
           li $v0, 10
13
                                 #exit
14
           syscall
    end main:
15
16
17
    sort:
           slt $s4,$a0,$a1
18
           beq $s4,$zero,done
                               #single element list is sorted
19
20
           addi $a0,$a0,4
           addi $s6,$a0,0
21
22
           j loop1
    after loop:
24
           la $tO,A
25
           la $t1,Aend
26 print:
           lw $at,0($t0)
27
28
           li $v0, 1
                                 # service 1 is print integer
29
           move $a0, $at
           syscall
                                 # execute
30
           li $v0, 4
31
           la $aO, Message1
32
           syscall
33
```

```
34
            addi $t0,$t0,4
35
            slt $t3,$t0,$t1
36
           bne $t3,$zero,print
37
           li $v0, 4
           la $aO, Message
38
            syscall
39
            addi $a0,$s6,0
                                   #$s6 = Address(A[i])
40
            sort
                                   #repeat sort
41
           j after sort
42
    done:
43
44
   loop1:
            addi $v0,$a0,0
                                   #init pointer to current element
45
46
           lw $v1,0($v0)
                                   #init value to current value
            addi $t0,$a0,0
                                   #init previous pointer to previous element
47
   loop:
48
           beq $t0,$s5,ret
49
                                    #if current=first, return
                                   #advance to previous element
            addi $t0,$t0,-4
50
           lw $t1,0($t0)
                                   #load previous element into $t1
51
52
            addi $s0,$v0,0
53
            addi $s1,$v1,0
54
            addi $v0,$t0,0
                                    #previous element is new current element
            addi $v1,$t1,0
                                    #privious value is new current value
55
56
            slt $t2,$t1,$s1
                                    #(previous) < (current) ?
           bne $t2,$zero,loop
                                   #if (previous) < (current), repeat
57
           lw $s3,0($t0)
                                    #
58
                                    #
            sw $s3,0($s0)
59
            sw $s1,0($t0)
                                    #swap 2 elements
60
            addi $v1,$s1,0
61
                                    #change completed; now repeat
           j loop
62
   ret:
63
            j after loop
64
```

