

Lab 6

Nguyễn Khánh Nam - 20225749

Assignment 1.....	1
Assignment 2.....	3
Assignment 4.....	5
Assignment 3.....	8

Assignment 1

Code:

.data

A: .word -12, 36, -1, 20, -2

.text

main:

```
    la    $a0,    A
    li    $a1,    5
    j     mspfx
    nop
```

continue:

```
lock: j     mspfx_end
      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 reaches.

@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:

```
    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:

```

    sll $t2, $t2, 2    #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:

```

Result:

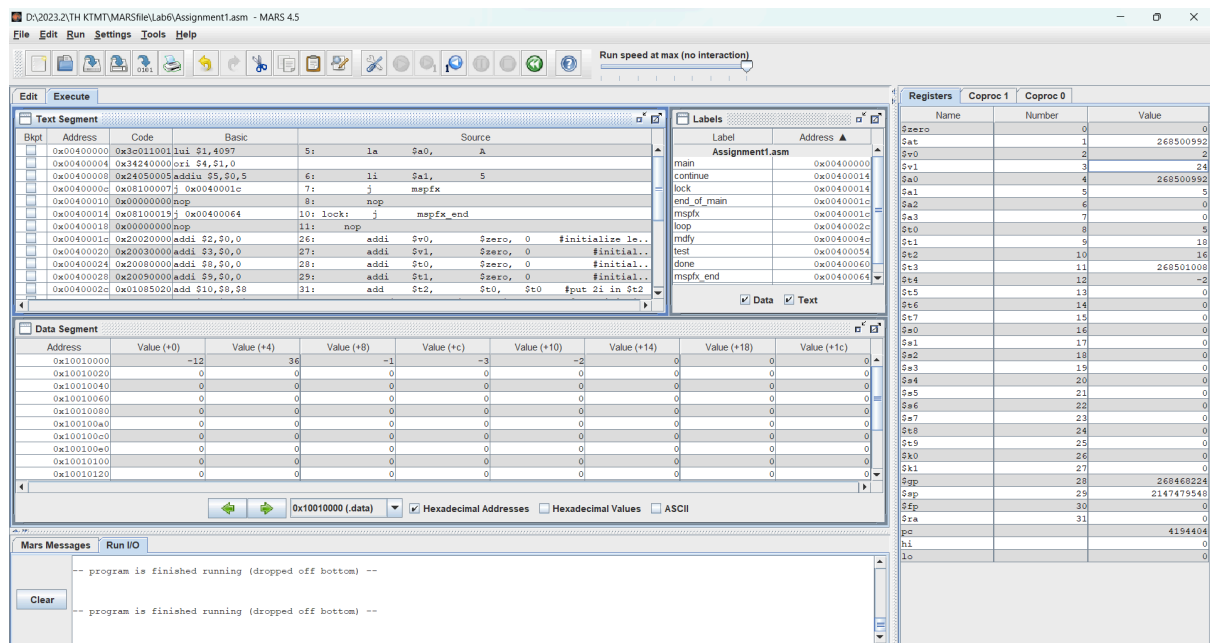
The screenshot displays the MARS 4.5 MIPS simulator interface. The main window is titled "D:\2023.2\TH KTM\T\MARSFile\Lab6\Assignment1.asm - MARS 4.5". The interface includes a menu bar (File, Edit, Run, Settings, Tools, Help), a toolbar, and a status bar indicating "Run speed at max (no interaction)".

The central pane is divided into three sections:

- Text Segment:** Displays assembly code with columns for Address, Code, Basic, and Source. The code includes instructions like `lui $t1, 4097`, `ori $t4, $t1, 0`, `addiu $t5, $t0, 5`, `li $t1, 5`, `nop`, `lock`, `lock: j mspfx_end`, `mspfx_end`, `addi $v0, $zero, 0`, `addi $v1, $zero, 0`, `addi $t0, $zero, 0`, `addi $t1, $zero, 0`, `addi $t2, $t0, $t0`, and `addi $t3, $t2, $t2`.
- Labels:** Lists labels and their addresses: `main` at `0x00400000`, `continue` at `0x00400014`, `lock` at `0x00400014`, `end_of_main` at `0x0040001c`, `mspfx` at `0x0040001c`, `loop` at `0x0040002c`, `mdfy` at `0x0040004c`, `test` at `0x00400054`, `done` at `0x00400060`, and `mspfx_end` at `0x00400064`.
- Data Segment:** Shows memory addresses and their values in decimal, hexadecimal, and ASCII formats. The values are mostly 0, with some non-zero values at the end of the segment.

On the right side, the **Registers** pane shows the state of MIPS registers. The `$zero` register is 0. The `$at` register is 1. The `$v0` register is 2. The `$v1` register is 3. The `$a0` register is 4. The `$a1` register is 5. The `$a2` register is 6. The `$a3` register is 7. The `$t0` register is 8. The `$t1` register is 9. The `$t2` register is 10. The `$t3` register is 11. The `$t4` register is 12. The `$t5` register is 13. The `$t6` register is 14. The `$t7` register is 15. The `$s0` register is 16. The `$s1` register is 17. The `$s2` register is 18. The `$s3` register is 19. The `$s4` register is 20. The `$s5` register is 21. The `$s6` register is 22. The `$s7` register is 23. The `$s8` register is 24. The `$s9` register is 25. The `$s0` register is 26. The `$s1` register is 27. The `$gp` register is 28. The `$sp` register is 29. The `$fp` register is 30. The `$ra` register is 31. The `pc` register is 4194304. The `hi` register is 0. The `lo` register is 0.

At the bottom, the **Mars Messages** pane shows the execution status. The message "Assembler: operation completed successfully." is displayed. The "Run I/O" button is visible.



Assignment 2

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:

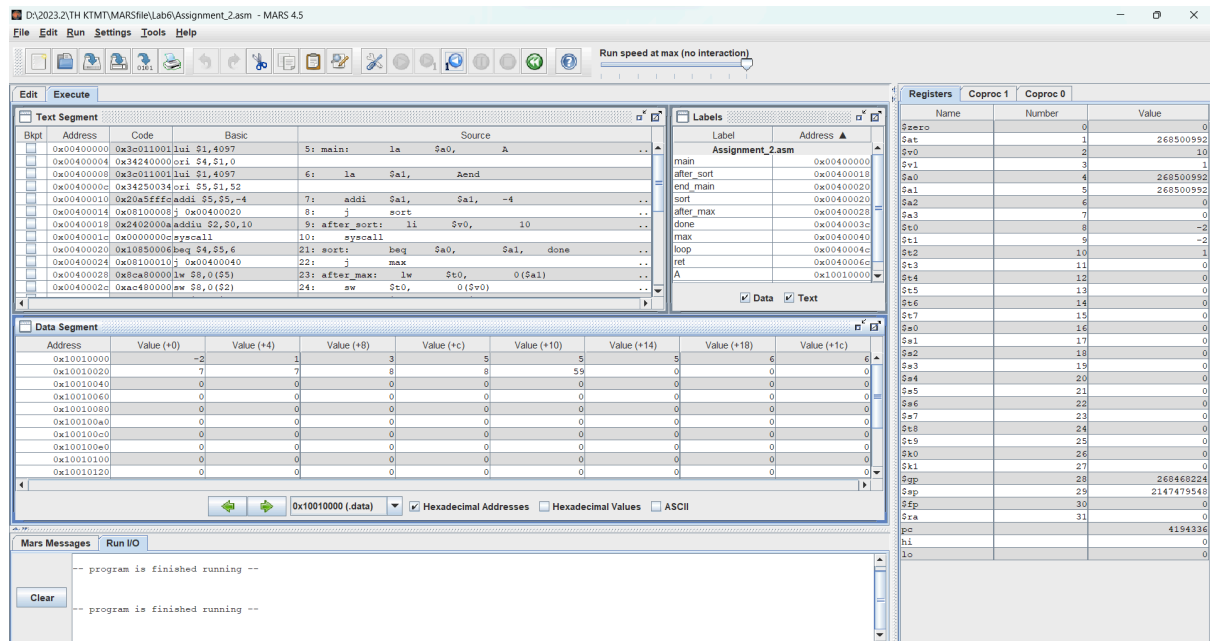
```

    j    after_max

```

Result:

The screenshot displays the MARS MIPS assembler simulator interface. The main window shows the assembly code for the 'max' procedure, which is being executed. The 'Text Segment' window displays the assembly code, and the 'Data Segment' window shows the memory layout. The 'Registers' window on the right shows the values of the registers, including \$zero, \$at, \$v0, \$v1, \$a0, \$a1, \$a2, \$a3, \$a4, \$a5, \$a6, \$a7, \$a8, \$a9, \$t0, \$t1, \$t2, \$t3, \$t4, \$t5, \$t6, \$t7, \$s0, \$s1, \$s2, \$s3, \$s4, \$s5, \$s6, \$s7, \$s8, \$s9, \$s10, \$s11, \$gp, \$sp, \$fp, \$ra, \$pc, \$hi, and \$lo. The 'Mars Messages' window at the bottom shows the execution progress, including the message 'pc: execution completed successfully.' and 'Assembly: operation completed successfully.'



Assignment 4

Code:

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

        /* Move elements of arr[0..i-1], that are
           greater than key, to one position ahead
           of their current position */
        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
```

#Lab 6, Assignment 3

.data

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

Aend: .word

.text

```
la $a0, A          #A address
addi $t0, $0, 13    # N
addi $t1, $t1, 1    #i=1
```

insertionSort:

```
slt $t2, $t1, $t0    #Kiem tra i < N
add $t3, $t1, -1     # j = i - 1
beq $t2, $0, end
nop
sll $t1, $t1, 2
add $a1, $a0, $t1    #Dia chi cua arr[i]
lw $s0, 0($a1)       #key = arr[i]
add $a2, $a1, -4     #Dia chi cua arr[j] voi j = i - 1
```

Swap:

```
lw $s1, 0($a2)       # = arr[j]
slt $t4, $t3, $0     #check j>=0
beq $t4, 1, End_swap
slt $t5, $s0, $s1    #check arr[j] > key
beq $t5, 0, End_swap
nop
sw $s1, 4($a2)        #Dia chi cua arr[j+1] va arr[j+1] = arr[j]
add $t3, $t3, -1     # j = j - 1
beq $t3, -1, End_swap #Check j<0 thi nhay ra khoi swap
nop
add $a2, $a2, -4     #Thay doi dia chi thanh arr[j] = j - 1]
j Swap
nop
```

End_swap:

```
add $t3, $t3, 1
sll $t3, $t3, 2      #Cap nhat lai j + 1
add $a2, $a0, $t3    #Dia chi cua arr[j+1]
sw $s0, 0($a2)       #arr[j+1] = key
srl $t1, $t1, 2
add $t1, $t1, 1
j insertionSort
nop
```

end:

Result:

D:\2023.2\TH KMTM\MARSfile\Lab6\Assignment_3.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	FA address
	0x00400000	0x3e011001	lui \$1,4097	0: la \$a0, A	
	0x00400004	0x34240000	ori \$4,\$1,0		
	0x00400008	0x20080000	addi \$0,\$0,13	9: addi \$t0, \$0, 13	# N
	0x0040000c	0x21290001	addi \$9,\$9,1	10: addi \$t1, \$t1, 1	#i=1
	0x00400010	0x0128502a	sllt \$10,\$9,\$8	13: sllt \$t2, \$t1, \$t0	#kiem tra i < N
	0x00400014	0x2128ffff	addi \$11,\$9,-1	14: add \$t3, \$t1, -1	# j = i - 1
	0x00400018	0x1140001d	beq \$10,\$9,\$9	15: beq \$t2, \$0, end	
	0x0040001c	0x00000000	nop	16: nop	
	0x00400020	0x00094880	all \$9,\$9,2	17: all \$t1, \$t1, 2	
	0x00400024	0x00092820	add \$5,\$4,\$9	18: add \$a1, \$a0, \$t1	#dia chi cua arr[i]
	0x00400028	0x8cb00000	lw \$16,0(\$5)	19: lw \$a0, 0(\$a1)	#key = arr[i]
	0x0040002c	0x20a6ffff	addi \$6,\$5,-4	20: add \$a2, \$a1, -4	#dia chi cua arr...

Labels

Label	Address
Assignment_3.asm	
InsertionSort	0x00400010
Swap	0x00400030
End_swap	0x00400070
end	0x00400090
A	0x10010000
Aend	0x10010034

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	9	9	5	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0

Mars Messages

Run I/O

Clear

Registers

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$a4	8	0
\$a5	9	0
\$a6	10	0
\$a7	11	0
\$a8	12	0
\$a9	13	0
\$t0	14	0
\$t1	15	0
\$t2	16	0
\$t3	17	0
\$t4	18	0
\$t5	19	0
\$t6	20	0
\$t7	21	0
\$t8	22	0
\$t9	23	0
\$t10	24	0
\$t11	25	0
\$t12	26	0
\$t13	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
pc		4194304
hi		0
lo		0

D:\2023.2\TH KMTM\MARSfile\Lab6\Assignment_3.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	FA address
	0x00400000	0x3e011001	lui \$1,4097	0: la \$a0, A	
	0x00400004	0x34240000	ori \$4,\$1,0		
	0x00400008	0x20080000	addi \$0,\$0,13	9: addi \$t0, \$0, 13	# N
	0x0040000c	0x21290001	addi \$9,\$9,1	10: addi \$t1, \$t1, 1	#i=1
	0x00400010	0x0128502a	sllt \$10,\$9,\$8	13: sllt \$t2, \$t1, \$t0	#kiem tra i < N
	0x00400014	0x2128ffff	addi \$11,\$9,-1	14: add \$t3, \$t1, -1	# j = i - 1
	0x00400018	0x1140001d	beq \$10,\$9,\$9	15: beq \$t2, \$0, end	
	0x0040001c	0x00000000	nop	16: nop	
	0x00400020	0x00094880	all \$9,\$9,2	17: all \$t1, \$t1, 2	
	0x00400024	0x00092820	add \$5,\$4,\$9	18: add \$a1, \$a0, \$t1	#dia chi cua arr[i]
	0x00400028	0x8cb00000	lw \$16,0(\$5)	19: lw \$a0, 0(\$a1)	#key = arr[i]
	0x0040002c	0x20a6ffff	addi \$6,\$5,-4	20: add \$a2, \$a1, -4	#dia chi cua arr...

Labels

Label	Address
Assignment_3.asm	
InsertionSort	0x00400010
Swap	0x00400030
End_swap	0x00400070
end	0x00400090
A	0x10010000
Aend	0x10010034

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	-2	1	8	5	5	6	7	3
0x10010020	7	7	8	8	5	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0

Mars Messages

Run I/O

Clear

Registers

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	268500992
\$a1	5	268501040
\$a2	6	268501012
\$a3	7	0
\$a4	8	13
\$a5	9	13
\$a6	10	0
\$a7	11	12
\$a8	12	0
\$a9	13	0
\$t0	14	0
\$t1	15	0
\$t2	16	5
\$t3	17	0
\$t4	18	0
\$t5	19	0
\$t6	20	0
\$t7	21	0
\$t8	22	0
\$t9	23	0
\$t10	24	0
\$t11	25	0
\$t12	26	0
\$t13	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
pc		4194448
hi		0
lo		0

Assignment 3

Code:

```
// Bubble sort
void BubbleSort(int arr[], int N)
{
    int i, j;
    for (i = 0; i < N; i++)
    {
        for (j = N-1; j > i; j--)
        {
            if(arr[j] < arr[j-1])
                swap(&arr[j], &arr[j-1]);
        }
    }
}
```

#Lab 6, Assignment 4

.data

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

Aend: .word

.text

la \$a0, A

addi \$t0, \$0, 13 # N

addi \$t1, \$0, 0 # i = 0

Loop1:

slt \$t2, \$t1, \$t0 #Check i < N

bne \$t2, 1, End_loop1

nop

addi \$t3, \$t0, -1 # j = N - 1

Loop2:

slt \$t4, \$t1, \$t3 #Check j > i

bne \$t4, 1, end_loop2


```

nop
sll $t3, $t3, 2
add $a1, $a0, $t3    #Dia chi cua arr[j]
lw $s0, 0($a1)       # = arr[j]
lw $s1, -4($a1)       # = arr[j-1]

slt $t5, $s0, $s1     #Check arr[j] < arr[j-1]
bne $t5, 1, if_loop2
nop
#SWAP
sw $s0, -4($a1)
sw $s1, 0($a1)
srl $t3, $t3, 2
addi $t3, $t3, -1     # j--
j Loop2
nop

```

end_loop2:

```

addi $t1, $t1, 1      # i++
j Loop1
nop

```

if_loop2:

```

srl $t3, $t3, 2
addi $t3, $t3, -1     # j--
j Loop2
nop

```

End_loop1:

Result:

D:\2023.2\TH KTM\T\MARSfile\Lab6\Assignment_4.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
	0x00400000	0x3e011000	lui \$1,4097	7: la \$a0, A
	0x00400004	0x34240000	ori \$4,\$1,0	
	0x00400008	0x20080000	addi \$0,\$0,13	8: addi \$t0, \$0, 13 # N
	0x0040000c	0x20090000	addi \$9,\$0,0	9: addi \$t1, \$0, 0 # i = 0
	0x00400010	0x0129502a	slt \$t0,\$t1,\$t0	12: slt \$t2, \$t1, \$t0 #check i < N
	0x00400014	0x20010001	addi \$1,\$0,1	13: bne \$t2, 1, End_loop1
	0x00400018	0x142a001b	bne \$1,\$10,27	
	0x0040001c	0x00000000	nop	14: nop
	0x00400020	0x210bffff	addi \$11,\$8,-1	15: addi \$t3, \$t0, -1 # j = N - 1
	0x00400024	0x012b602a	slt \$t2,\$t1,\$t3	18: slt \$t4, \$t1, \$t3 #check j > i
	0x00400028	0x20010001	addi \$1,\$0,1	19: bne \$t4, 1, end_loop2
	0x0040002c	0x142c000f	bne \$1,\$12,15	

Labels

Label	Address
Assignment_4.asm	
Loop1	0x00400010
Loop2	0x00400024
end_loop2	0x0040006c
f_loop2	0x00400078
End_loop1	0x00400088
A	0x10010000
Aend	0x10010034

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	9	5	5	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0

Mars Messages Run I/O

Assemble: assembling D:\2023.2\TH KTM\T\MARSfile\Lab6\Assignment_4.asm

Assemble: operation completed successfully.

Registers Coproc 1 Coproc 0

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$a0	8	0
\$t0	9	0
\$t2	10	0
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$a0	16	0
\$a1	17	0
\$a2	18	0
\$a3	19	0
\$a4	20	0
\$a5	21	0
\$a6	22	0
\$a7	23	0
\$t8	24	0
\$t9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$fp	29	2147479548
\$ra	31	0
\$pc		4194304
\$hi		0
\$lo		0

D:\2023.2\TH KTM\T\MARSfile\Lab6\Assignment_4.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
	0x00400000	0x3e011000	lui \$1,4097	7: la \$a0, A
	0x00400004	0x34240000	ori \$4,\$1,0	
	0x00400008	0x20080000	addi \$0,\$0,13	8: addi \$t0, \$0, 13 # N
	0x0040000c	0x20090000	addi \$9,\$0,0	9: addi \$t1, \$0, 0 # i = 0
	0x00400010	0x0129502a	slt \$t0,\$t1,\$t0	12: slt \$t2, \$t1, \$t0 #check i < N
	0x00400014	0x20010001	addi \$1,\$0,1	13: bne \$t2, 1, End_loop1
	0x00400018	0x142a001b	bne \$1,\$10,27	
	0x0040001c	0x00000000	nop	14: nop
	0x00400020	0x210bffff	addi \$11,\$8,-1	15: addi \$t3, \$t0, -1 # j = N - 1
	0x00400024	0x012b602a	slt \$t2,\$t1,\$t3	18: slt \$t4, \$t1, \$t3 #check j > i
	0x00400028	0x20010001	addi \$1,\$0,1	19: bne \$t4, 1, end_loop2
	0x0040002c	0x142c000f	bne \$1,\$12,15	

Labels

Label	Address
Assignment_4.asm	
Loop1	0x00400010
Loop2	0x00400024
end_loop2	0x0040006c
f_loop2	0x00400078
End_loop1	0x00400088
A	0x10010000
Aend	0x10010034

Data Segment

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	3
0x10010020	7	7	8	8	5	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0

Mars Messages Run I/O

Reset: reset completed.

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

Registers Coproc 1 Coproc 0

Name	Number	Value
\$zero	0	0
\$at	1	1
\$v0	2	0
\$v1	3	0
\$a0	4	268500992
\$a1	5	268501040
\$a2	6	0
\$a3	7	0
\$a0	8	13
\$t0	9	0
\$t2	10	0
\$t3	11	12
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$a0	16	59
\$a1	17	8
\$a2	18	0
\$a3	19	0
\$a4	20	0
\$a5	21	0
\$a6	22	0
\$a7	23	0
\$t8	24	0
\$t9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$fp	29	2147479548
\$ra	31	0
\$pc		4194440
\$hi		0
\$lo		0