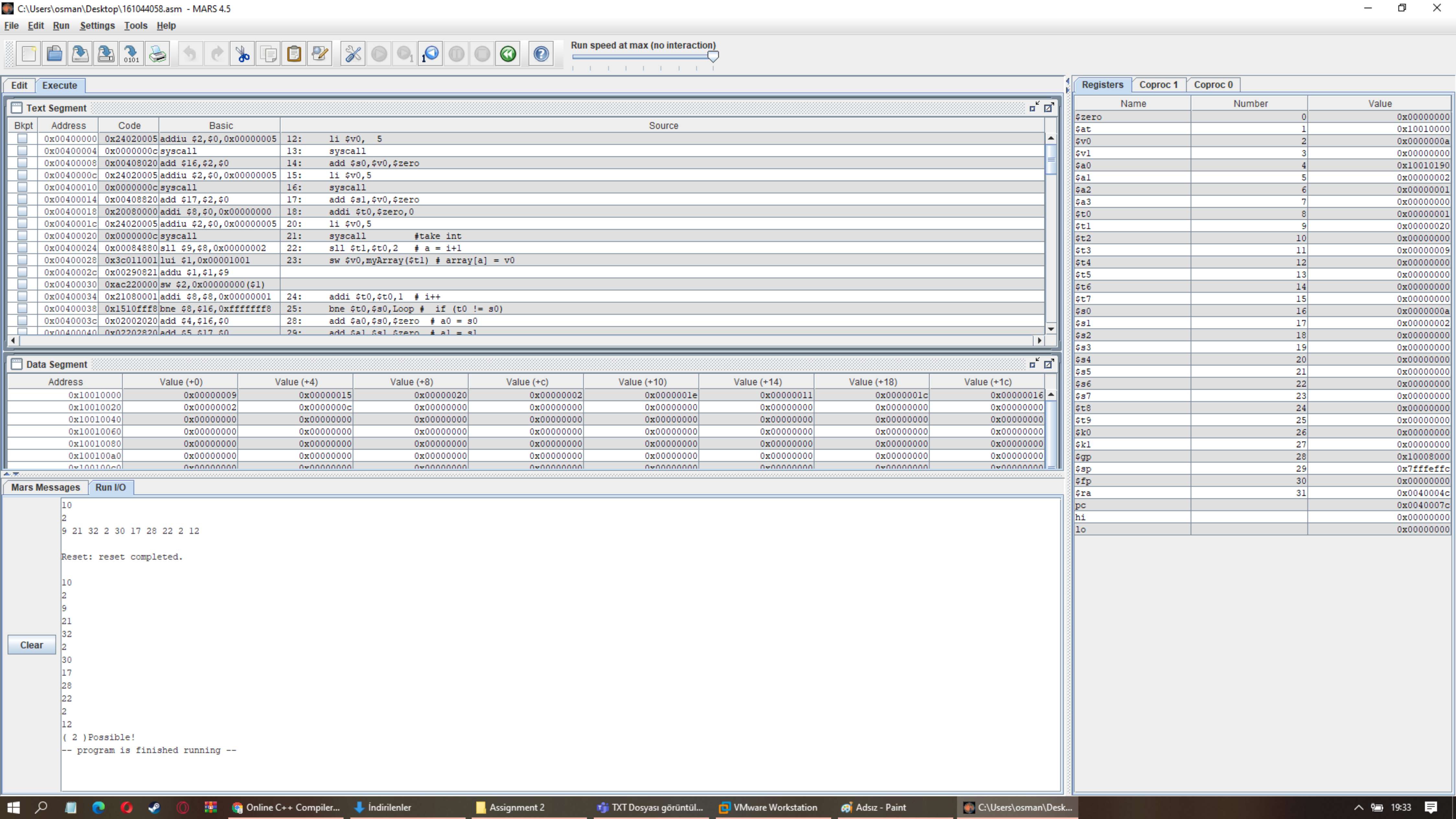


C:\Users\osman\Desktop\161044058.asm - MARS 4.5

Text Segment

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x24020005	addiu $2,$0,0x00000005	12: li $v0, 5
□	0x00400004	0x0000000c	syscall	13: syscall
□	0x00400008	0x00408020	add $16,$2,$0	14: add $s0,$v0,$zero
□	0x0040000c	0x24020005	addiu $2,$0,0x00000005	15: li $v0,5
□	0x00400010	0x0000000c	syscall	16: syscall
□	0x00400014	0x00408820	add $17,$2,$0	17: add $s1,$v0,$zero
□	0x00400018	0x20080000	addi $8,$0,0x00000000	18: addi $t0,$zero,0
□	0x0040001c	0x24020005	addiu $2,$0,0x00000005	20: li $v0,5
□	0x00400020	0x0000000c	syscall	21: syscall #take int
□	0x00400024	0x00084880	sll $9,$8,0x00000002	22: sll $t1,$t0,2 # a = i+1
□	0x00400028	0x3c011001	lui $1,0x00001001	23: sw $v0,myArray($t1) # array[a] = v0
□	0x0040002c	0x00290821	addu $1,$1,$9	
□	0x00400030	0xac220000	sw $2,0x00000000($1)	
□	0x00400034	0x21080001	addi $8,$8,0x00000001	24: addi $t0,$t0,1 # i++
□	0x00400038	0x1510fff8	bne $8,$16,0xfffffff8	25: bne $t0,$s0,Loop # if (t0 != s0)
□	0x0040003c	0x02002020	add $4,$16,$0	28: add $a0,$s0,$zero # a0 = s0
□	0x00400040	0x02202820	add $5,$17,$0	29: add $a1,$s1,$zero # a1 = s1

Data Segment

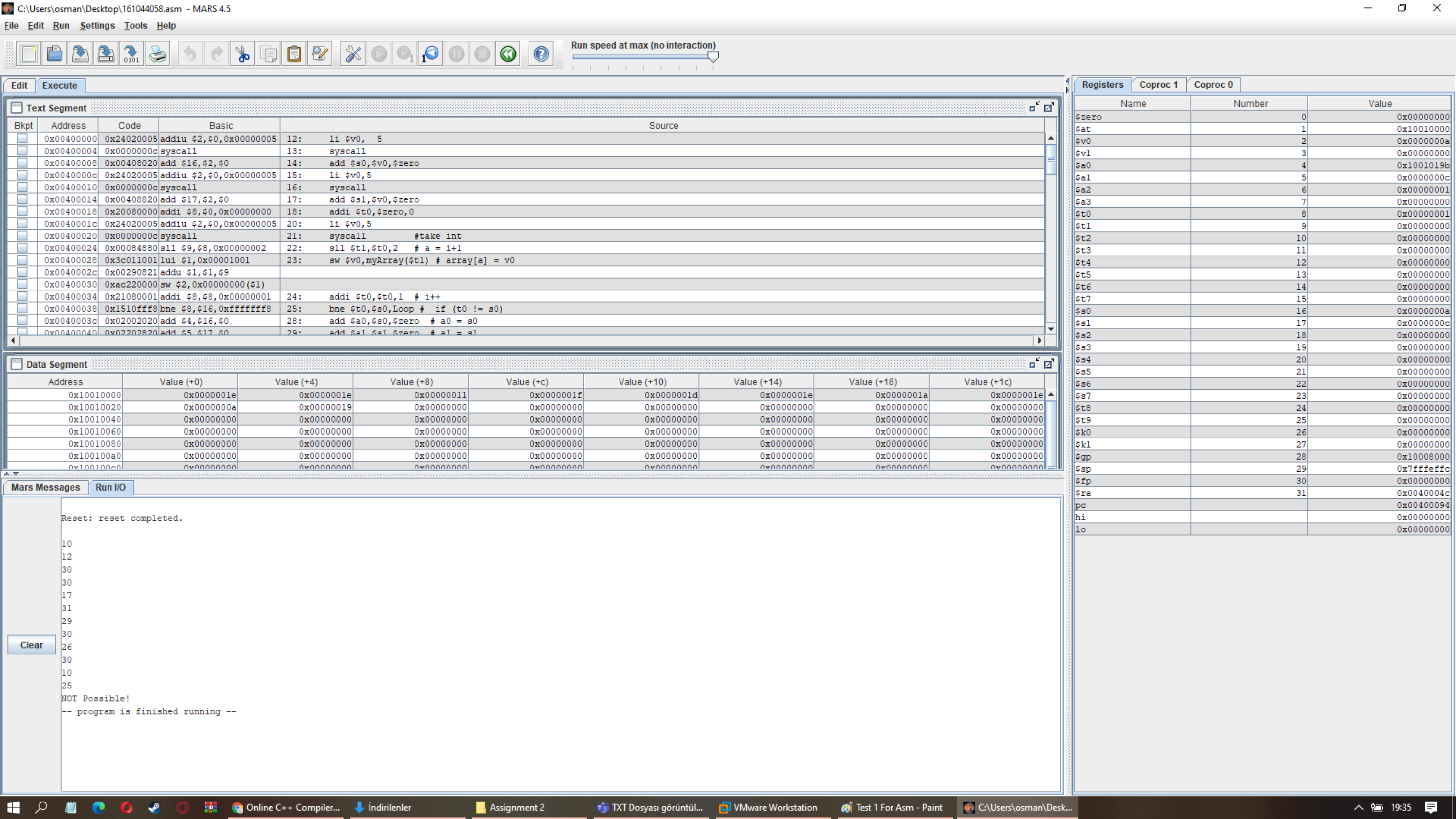
Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000009	0x00000015	0x00000020	0x00000002	0x0000001e	0x00000011	0x0000001c	0x00000016
0x10010020	0x00000002	0x0000000c	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

```
10
2
9 21 32 2 30 17 28 22 2 12

Reset: reset completed.

10
2
9
21
32
2
30
17
28
22
2
12
( 2 )Possible!
-- program is finished running --
```

Name	Number	Value
$zero	0	0x00000000
$at	1	0x10010000
$v0	2	0x0000000a
$v1	3	0x00000000
$a0	4	0x10010190
$a1	5	0x00000002
$a2	6	0x00000001
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x00000020
$t2	10	0x00000000
$t3	11	0x00000009
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0000000a
$s1	17	0x00000002
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x0040004c
pc		0x0040007c
hi		0x00000000
lo		0x00000000



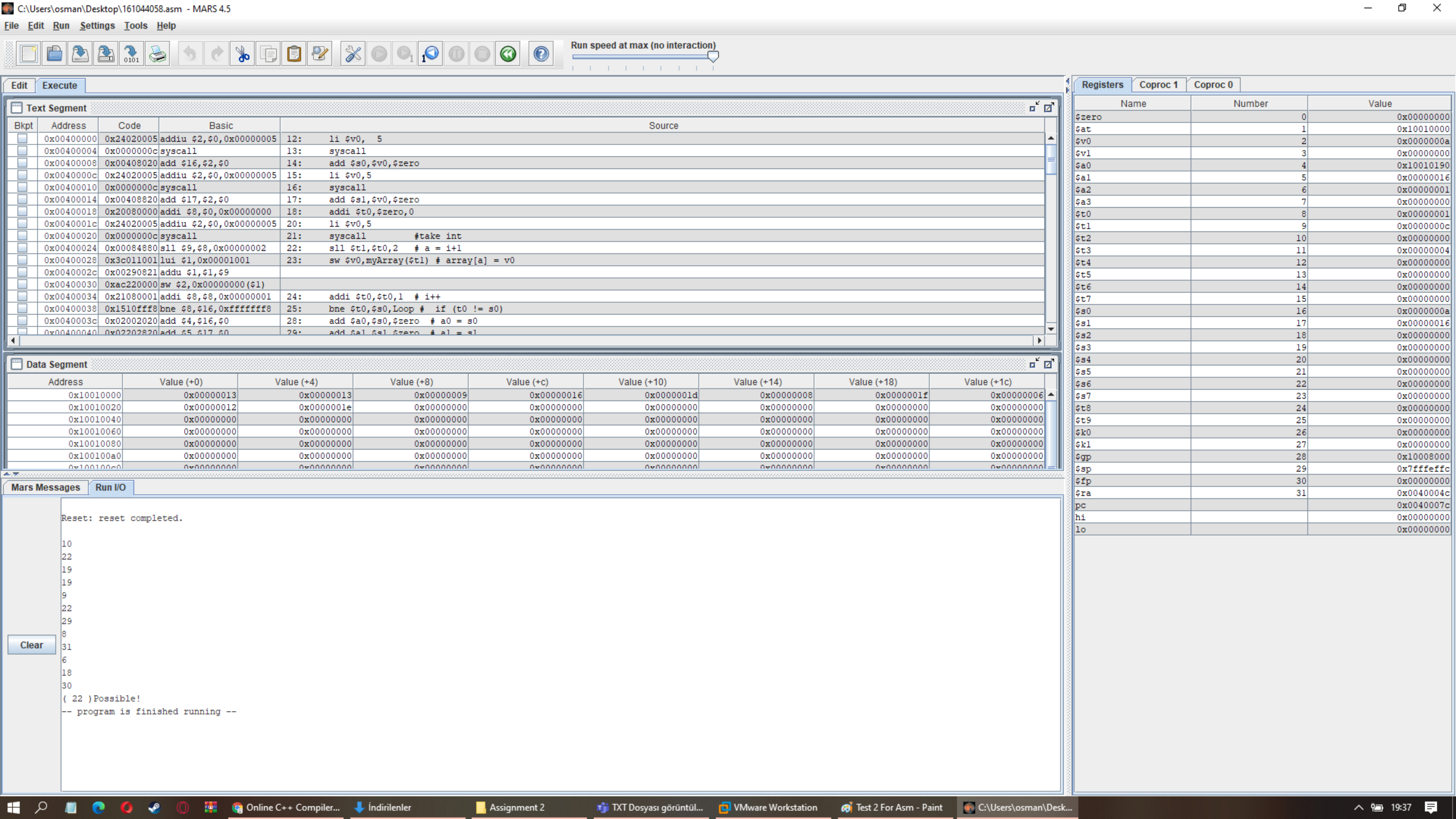
Bkpt	Address	Code	Basic	Source
□	0x00400000	0x24020005	addiu $2,$0,0x00000005	12: li $v0, 5
□	0x00400004	0x0000000c	syscall	13: syscall
□	0x00400008	0x00408020	add $16,$2,$0	14: add $s0,$v0,$zero
□	0x0040000c	0x24020005	addiu $2,$0,0x00000005	15: li $v0,5
□	0x00400010	0x0000000c	syscall	16: syscall
□	0x00400014	0x00408820	add $17,$2,$0	17: add $s1,$v0,$zero
□	0x00400018	0x20080000	addi $8,$0,0x00000000	18: addi $t0,$zero,0
□	0x0040001c	0x24020005	addiu $2,$0,0x00000005	20: li $v0,5
□	0x00400020	0x0000000c	syscall	21: syscall #take int
□	0x00400024	0x00084880	sll $9,$8,0x00000002	22: sll $t1,$t0,2 # a = i+1
□	0x00400028	0x3c011001	lui $1,0x00001001	23: sw $v0,myArray($t1) # array[a] = v0
□	0x0040002c	0x00290821	addu $1,$1,$9	
□	0x00400030	0xac220000	sw $2,0x00000000($1)	
□	0x00400034	0x21080001	addi $8,$8,0x00000001	24: addi $t0,$t0,1 # i++
□	0x00400038	0x1510fff8	bne $8,$16,0xfffffff8	25: bne $t0,$s0,Loop # if (t0 != s0)
□	0x0040003c	0x02002020	add $4,$16,$0	28: add $a0,$s0,$zero # a0 = s0
□	0x00400040	0x02202820	add $5,$17,$0	29: add $a1,$s1,$zero # a1 = s1

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x0000001e	0x0000001e	0x00000011	0x0000001f	0x0000001d	0x0000001e	0x0000001a	0x0000001e
0x10010020	0x0000000a	0x00000019	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

```
Reset: reset completed.

10
12
30
30
17
31
29
30
26
30
10
25
NOT Possible!
-- program is finished running --
```

Name	Number	Value
$zero	0	0x00000000
$at	1	0x10010000
$v0	2	0x0000000a
$v1	3	0x00000000
$a0	4	0x1001019b
$a1	5	0x0000000c
$a2	6	0x00000001
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x00000000
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0000000a
$s1	17	0x0000000c
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x0040004c
pc		0x00400094
hi		0x00000000
lo		0x00000000



C:\Users\osman\Desktop\161044058.asm - MARS 4.5

Run speed at max (no interaction)

Text Segment

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x24020005	addiu $2,$0,0x00000005	12: li $v0, 5
□	0x00400004	0x0000000c	syscall	13: syscall
□	0x00400008	0x00408020	add $16,$2,$0	14: add $s0,$v0,$zero
□	0x0040000c	0x24020005	addiu $2,$0,0x00000005	15: li $v0,5
□	0x00400010	0x0000000c	syscall	16: syscall
□	0x00400014	0x00408820	add $17,$2,$0	17: add $s1,$v0,$zero
□	0x00400018	0x20080000	addi $8,$0,0x00000000	18: addi $t0,$zero,0
□	0x0040001c	0x24020005	addiu $2,$0,0x00000005	20: li $v0,5
□	0x00400020	0x0000000c	syscall	21: syscall #take int
□	0x00400024	0x00084880	sll $9,$8,0x00000002	22: sll $t1,$t0,2 # a = i+1
□	0x00400028	0x3c011001	lui $1,0x00001001	23: sw $v0,myArray($t1) # array[a] = v0
□	0x0040002c	0x00290821	addu $1,$1,$9	
□	0x00400030	0xac220000	sw $2,0x00000000($1)	
□	0x00400034	0x21080001	addi $8,$8,0x00000001	24: addi $t0,$t0,1 # i++
□	0x00400038	0x1510fff8	bne $8,$16,0xfffffff8	25: bne $t0,$s0,Loop # if (t0 != s0)
□	0x0040003c	0x02002020	add $4,$16,$0	28: add $a0,$s0,$zero # a0 = s0
□	0x00400040	0x02202820	add $5,$17,$0	29: add $a1,$s1,$zero # a1 = s1

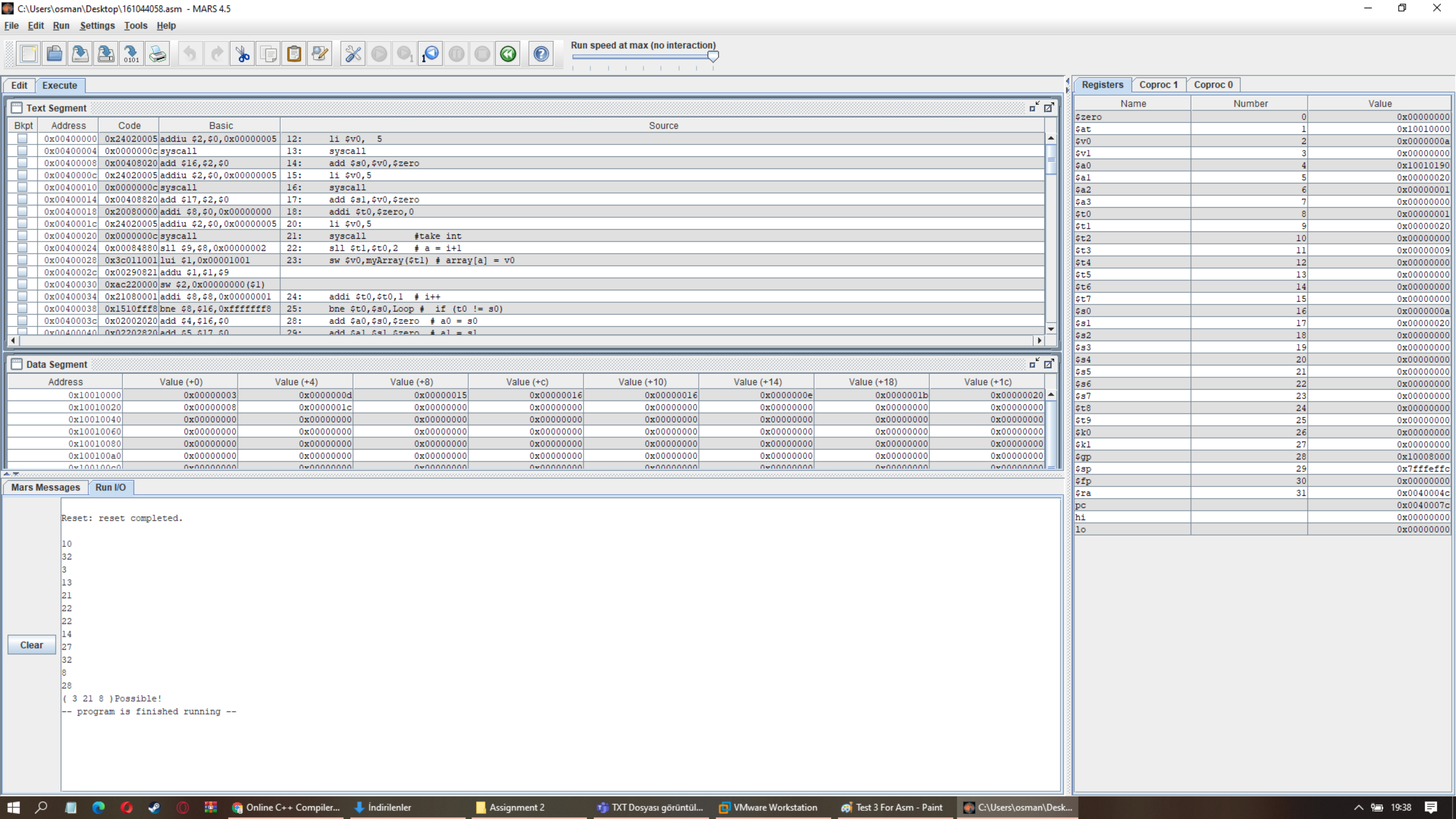
Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000013	0x00000013	0x00000009	0x00000016	0x0000001d	0x00000008	0x0000001f	0x00000006
0x10010020	0x00000012	0x0000001e	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

```
Reset: reset completed.

10
22
19
19
9
22
29
8
31
6
18
30
( 22 )Possible!
-- program is finished running --
```

Name	Number	Value
$zero	0	0x00000000
$at	1	0x10010000
$v0	2	0x0000000a
$v1	3	0x00000000
$a0	4	0x10010190
$a1	5	0x00000016
$a2	6	0x00000001
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x0000000c
$t2	10	0x00000000
$t3	11	0x00000004
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0000000a
$s1	17	0x00000016
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x0040004c
pc		0x0040007c
hi		0x00000000
lo		0x00000000



Text Segment

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x24020005	addiu $2,$0,0x00000005	12: li $v0, 5
□	0x00400004	0x0000000c	syscall	13: syscall
□	0x00400008	0x00408020	add $16,$2,$0	14: add $s0,$v0,$zero
□	0x0040000c	0x24020005	addiu $2,$0,0x00000005	15: li $v0,5
□	0x00400010	0x0000000c	syscall	16: syscall
□	0x00400014	0x00408820	add $17,$2,$0	17: add $s1,$v0,$zero
□	0x00400018	0x20080000	addi $8,$0,0x00000000	18: addi $t0,$zero,0
□	0x0040001c	0x24020005	addiu $2,$0,0x00000005	20: li $v0,5
□	0x00400020	0x0000000c	syscall	21: syscall #take int
□	0x00400024	0x00084880	sll $9,$8,0x00000002	22: sll $t1,$t0,2 # a = i+1
□	0x00400028	0x3c011001	lui $1,0x00001001	23: sw $v0,myArray($t1) # array[a] = v0
□	0x0040002c	0x00290821	addu $1,$1,$9	
□	0x00400030	0xac220000	sw $2,0x00000000($1)	
□	0x00400034	0x21080001	addi $8,$8,0x00000001	24: addi $t0,$t0,1 # i++
□	0x00400038	0x1510fff8	bne $8,$16,0xfffffff8	25: bne $t0,$s0,Loop # if (t0 != s0)
□	0x0040003c	0x02002020	add $4,$16,$0	28: add $a0,$s0,$zero # a0 = s0
□	0x00400040	0x02202820	add $5,$17,$0	29: add $a1,$s1,$zero # a1 = s1

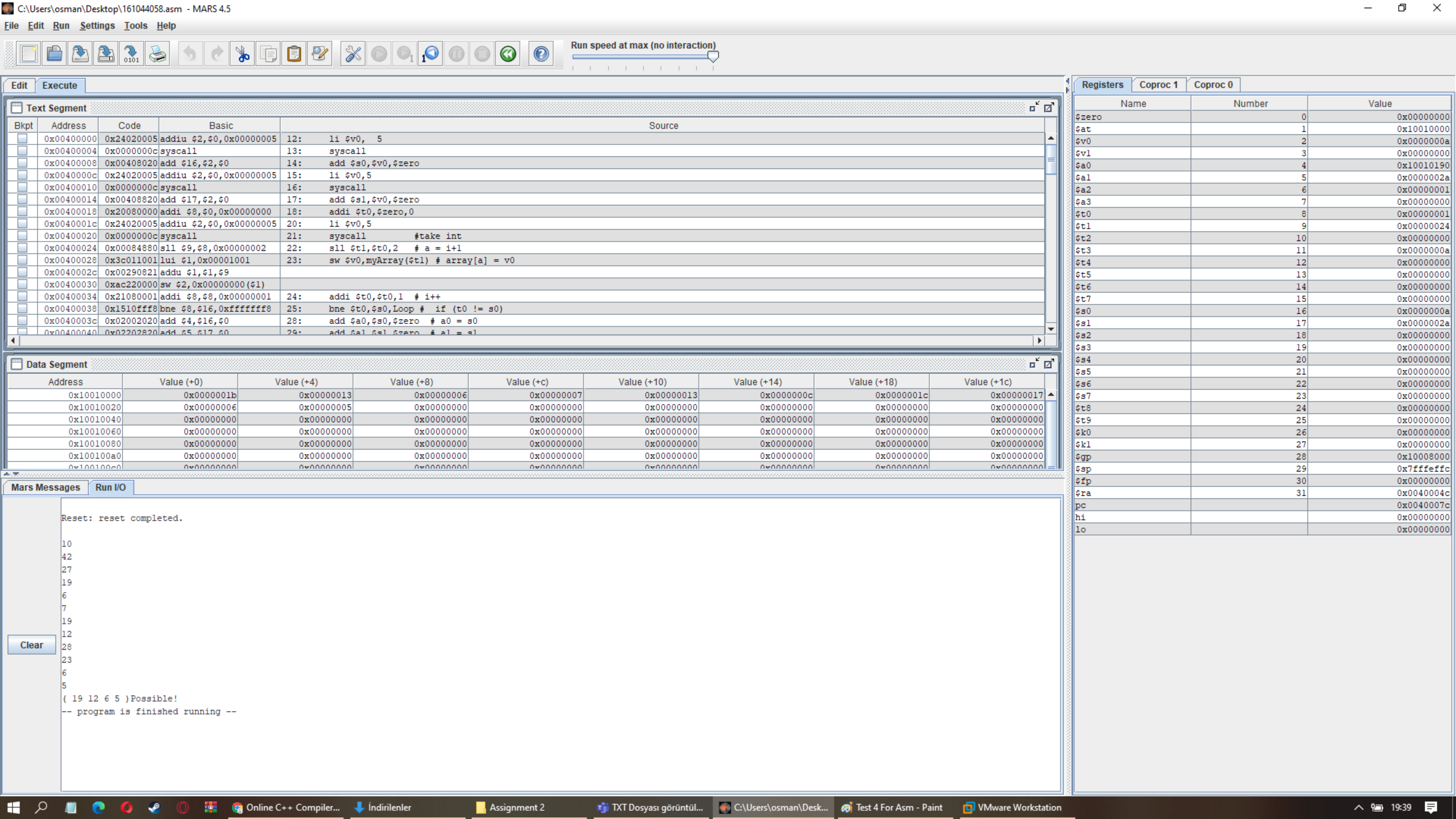
Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000003	0x0000000d	0x00000015	0x00000016	0x00000016	0x0000000e	0x0000001b	0x00000020
0x10010020	0x00000008	0x0000001c	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

```
Reset: reset completed.

10
32
3
13
21
22
22
14
27
32
8
28
( 3 21 8 )Possible!
-- program is finished running --
```

Name	Number	Value
$zero	0	0x00000000
$at	1	0x10010000
$v0	2	0x0000000a
$v1	3	0x00000000
$a0	4	0x10010190
$a1	5	0x00000020
$a2	6	0x00000001
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x00000020
$t2	10	0x00000000
$t3	11	0x00000009
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0000000a
$s1	17	0x00000020
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x0040004c
pc		0x0040007c
hi		0x00000000
lo		0x00000000



Text Segment

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x24020005	addiu $2,$0,0x00000005	12: li $v0, 5
□	0x00400004	0x0000000c	syscall	13: syscall
□	0x00400008	0x00408020	add $16,$2,$0	14: add $s0,$v0,$zero
□	0x0040000c	0x24020005	addiu $2,$0,0x00000005	15: li $v0,5
□	0x00400010	0x0000000c	syscall	16: syscall
□	0x00400014	0x00408820	add $17,$2,$0	17: add $s1,$v0,$zero
□	0x00400018	0x20080000	addi $8,$0,0x00000000	18: addi $t0,$zero,0
□	0x0040001c	0x24020005	addiu $2,$0,0x00000005	20: li $v0,5
□	0x00400020	0x0000000c	syscall	21: syscall #take int
□	0x00400024	0x00084880	sll $9,$8,0x00000002	22: sll $t1,$t0,2 # a = i+1
□	0x00400028	0x3c011001	lui $1,0x00001001	23: sw $v0,myArray($t1) # array[a] = v0
□	0x0040002c	0x00290821	addu $1,$1,$9	
□	0x00400030	0xac220000	sw $2,0x00000000($1)	
□	0x00400034	0x21080001	addi $8,$8,0x00000001	24: addi $t0,$t0,1 # i++
□	0x00400038	0x1510fff8	bne $8,$16,0xfffffff8	25: bne $t0,$s0,Loop # if (t0 != s0)
□	0x0040003c	0x02002020	add $4,$16,$0	28: add $a0,$s0,$zero # a0 = s0
□	0x00400040	0x02202820	add $5,$17,$0	29: add $a1,$s1,$zero # a1 = s1

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x0000001b	0x00000013	0x00000006	0x00000007	0x00000013	0x0000000c	0x0000001c	0x00000017
0x10010020	0x00000006	0x00000005	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

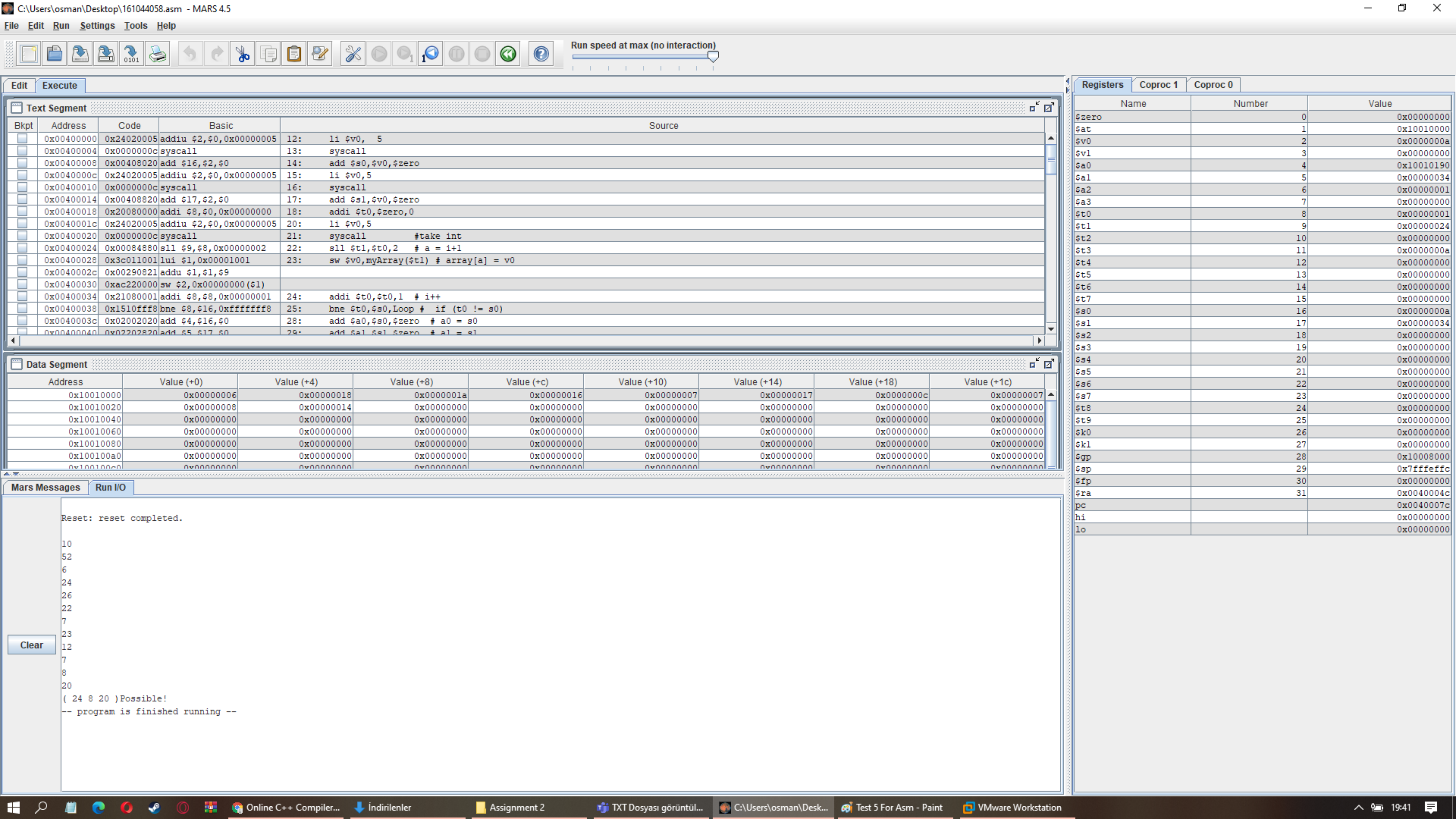
Run I/O

```
Reset: reset completed.

10
42
27
19
6
7
19
12
28
23
6
5
( 19 12 6 5 )Possible!
-- program is finished running --
```

Registers

Name	Number	Value
$zero	0	0x00000000
$at	1	0x10010000
$v0	2	0x0000000a
$v1	3	0x00000000
$a0	4	0x10010190
$a1	5	0x0000002a
$a2	6	0x00000001
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x00000024
$t2	10	0x00000000
$t3	11	0x0000000a
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0000000a
$s1	17	0x0000002a
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x0040004c
pc		0x0040007c
hi		0x00000000
lo		0x00000000



C:\Users\osman\Desktop\161044058.asm - MARS 4.5

Text Segment

Bkpt	Address	Code	Basic	Source
□	0x00400000	0x24020005	addiu $2,$0,0x00000005	12: li $v0, 5
□	0x00400004	0x0000000c	syscall	13: syscall
□	0x00400008	0x00408020	add $16,$2,$0	14: add $s0,$v0,$zero
□	0x0040000c	0x24020005	addiu $2,$0,0x00000005	15: li $v0,5
□	0x00400010	0x0000000c	syscall	16: syscall
□	0x00400014	0x00408820	add $17,$2,$0	17: add $s1,$v0,$zero
□	0x00400018	0x20080000	addi $8,$0,0x00000000	18: addi $t0,$zero,0
□	0x0040001c	0x24020005	addiu $2,$0,0x00000005	20: li $v0,5
□	0x00400020	0x0000000c	syscall	21: syscall #take int
□	0x00400024	0x00084880	sll $9,$8,0x00000002	22: sll $t1,$t0,2 # a = i+1
□	0x00400028	0x3c011001	lui $1,0x00001001	23: sw $v0,myArray($t1) # array[a] = v0
□	0x0040002c	0x00290821	addu $1,$1,$9	
□	0x00400030	0xac220000	sw $2,0x00000000($1)	
□	0x00400034	0x21080001	addi $8,$8,0x00000001	24: addi $t0,$t0,1 # i++
□	0x00400038	0x1510fff8	bne $8,$16,0xfffffff8	25: bne $t0,$s0,Loop # if (t0 != s0)
□	0x0040003c	0x02002020	add $4,$16,$0	28: add $a0,$s0,$zero # a0 = s0
□	0x00400040	0x02202820	add $5,$17,$0	29: add $a1,$s1,$zero # a1 = s1

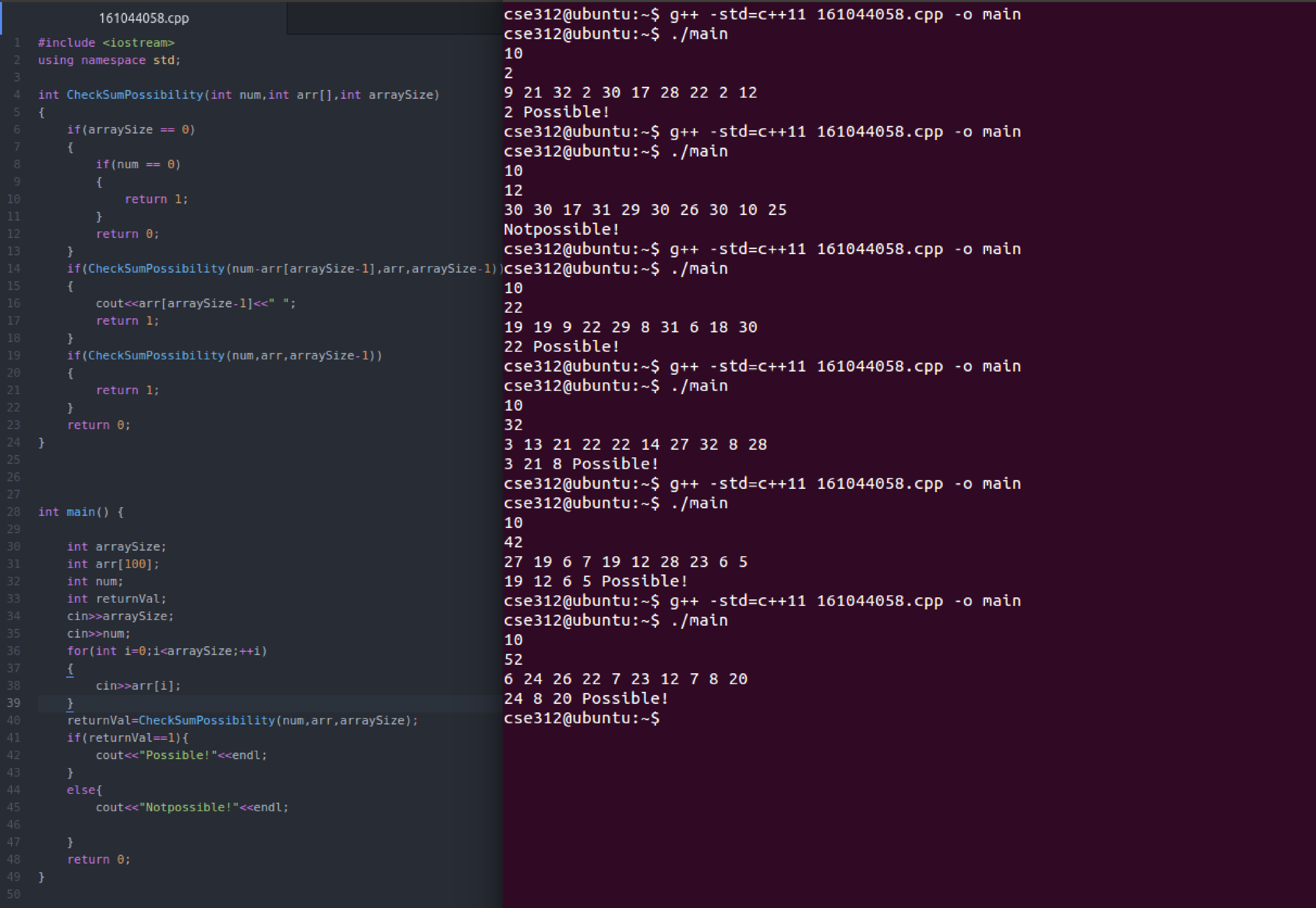
Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000006	0x00000018	0x0000001a	0x00000016	0x00000007	0x00000017	0x0000000c	0x00000007
0x10010020	0x00000008	0x00000014	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

```
Reset: reset completed.

10
52
6
24
26
22
7
23
12
7
8
20
( 24 8 20 )Possible!
-- program is finished running --
```

Name	Number	Value
$zero	0	0x00000000
$at	1	0x10010000
$v0	2	0x0000000a
$v1	3	0x00000000
$a0	4	0x10010190
$a1	5	0x00000034
$a2	6	0x00000001
$a3	7	0x00000000
$t0	8	0x00000001
$t1	9	0x00000024
$t2	10	0x00000000
$t3	11	0x0000000a
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x0000000a
$s1	17	0x00000034
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x0040004c
pc		0x0040007c
hi		0x00000000
lo		0x00000000



```cpp
#include <iostream>
using namespace std;

int CheckSumPossibility(int num,int arr[],int arraySize)
{
    if(arraySize == 0)
    {
        if(num == 0)
        {
            return 1;
        }
        return 0;
    }
    if(CheckSumPossibility(num-arr[arraySize-1],arr,arraySize-1))
    {
        cout<<arr[arraySize-1]<<" ";
        return 1;
    }
    if(CheckSumPossibility(num,arr,arraySize-1))
    {
        return 1;
    }
    return 0;
}



int main() {

    int arraySize;
    int arr[100];
    int num;
    int returnVal;
    cin>>arraySize;
    cin>>num;
    for(int i=0;i<arraySize;++i)
    {
        cin>>arr[i];
    }
    returnVal=CheckSumPossibility(num,arr,arraySize);
    if(returnVal==1){
        cout<<"Possible!"<<endl;
    }
    else{
        cout<<"Notpossible!"<<endl;

    }
    return 0;
}
```

```
cse312@ubuntu:~$ g++ -std=c++11 161044058.cpp -o main
cse312@ubuntu:~$ ./main
10
2
9 21 32 2 30 17 28 22 2 12
2 Possible!
cse312@ubuntu:~$ g++ -std=c++11 161044058.cpp -o main
cse312@ubuntu:~$ ./main
10
12
30 30 17 31 29 30 26 30 10 25
Notpossible!
cse312@ubuntu:~$ g++ -std=c++11 161044058.cpp -o main
cse312@ubuntu:~$ ./main
10
22
19 19 9 22 29 8 31 6 18 30
22 Possible!
cse312@ubuntu:~$ g++ -std=c++11 161044058.cpp -o main
cse312@ubuntu:~$ ./main
10
32
3 13 21 22 22 14 27 32 8 28
3 21 8 Possible!
cse312@ubuntu:~$ g++ -std=c++11 161044058.cpp -o main
cse312@ubuntu:~$ ./main
10
42
27 19 6 7 19 12 28 23 6 5
19 12 6 5 Possible!
cse312@ubuntu:~$ g++ -std=c++11 161044058.cpp -o main
cse312@ubuntu:~$ ./main
10
52
6 24 26 22 7 23 12 7 8 20
24 8 20 Possible!
cse312@ubuntu:~$
```

Bonus Part:

My program can print the number.

Explanation for assembly register:

- I used $s0,$s1
  - Ss0 = size
  - $s1 = num
- I used $a0,$a1,$a2 like a procedure parameter
  - $a0 = size
  - $a1 = num
- $a2 = 1 // I used $a2 like a controller for if statement.
- I used $t registers for temporary value
- I used $sp register for save the $ra return address
- I used $v0 register , This register takes return value

Note:

I write "CheckSumPossibility" as a procedure but this procedure name is "sub".

I didn't optimize my code. I said that fort this suggestion "you must ignore the next recursive calls when the sum exceeded the target number num".