**Bahria University**

**Software Engineering Department**



**Course: CEL-221 Computer Architecture and Organization**

**Term: Fall 2019, Class: BSE 3(B)**

**Assignment No:**

|  |  |
| --- | --- |
| **0** | **1** |

**Submitted By:**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Q** | **A** | **I** | **S** | **E** | **R** |  | **A** | **B** | **B** | **A** | **S** |

**Enrollment No.:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 2 | - | 1 | 3 | 1 | 1 | 8 | 2 | - | 0 | 3 | 0 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Submission Date** | | | | | | | |  |
| **3** | **1** | **/** | **1** | **2** | **/** | **1** | **9** |

**Submitted To:**

**Engr. Zohaib Shahzad**

**Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Max Marks: \_\_\_\_\_\_\_\_\_\_\_ Marks Obtained: \_\_\_\_\_\_\_\_\_\_\_\_\_**

**Q:- *Find 10 instructions of MIPS that are not studied in class and also write 1 example for each instruction?***

# slt $t1,$t2,$t3

**Set less than : If $t2 is less than $t3, then set $t1 to 1 else set $t1 to 0**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "\nOutput we are comparing input val with 3\nAfter performing Set less than operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

li $t2,4

slt $t1,$t0,$t2 #set less than

move $a0,$t1

la $a0,output

li $v0,4

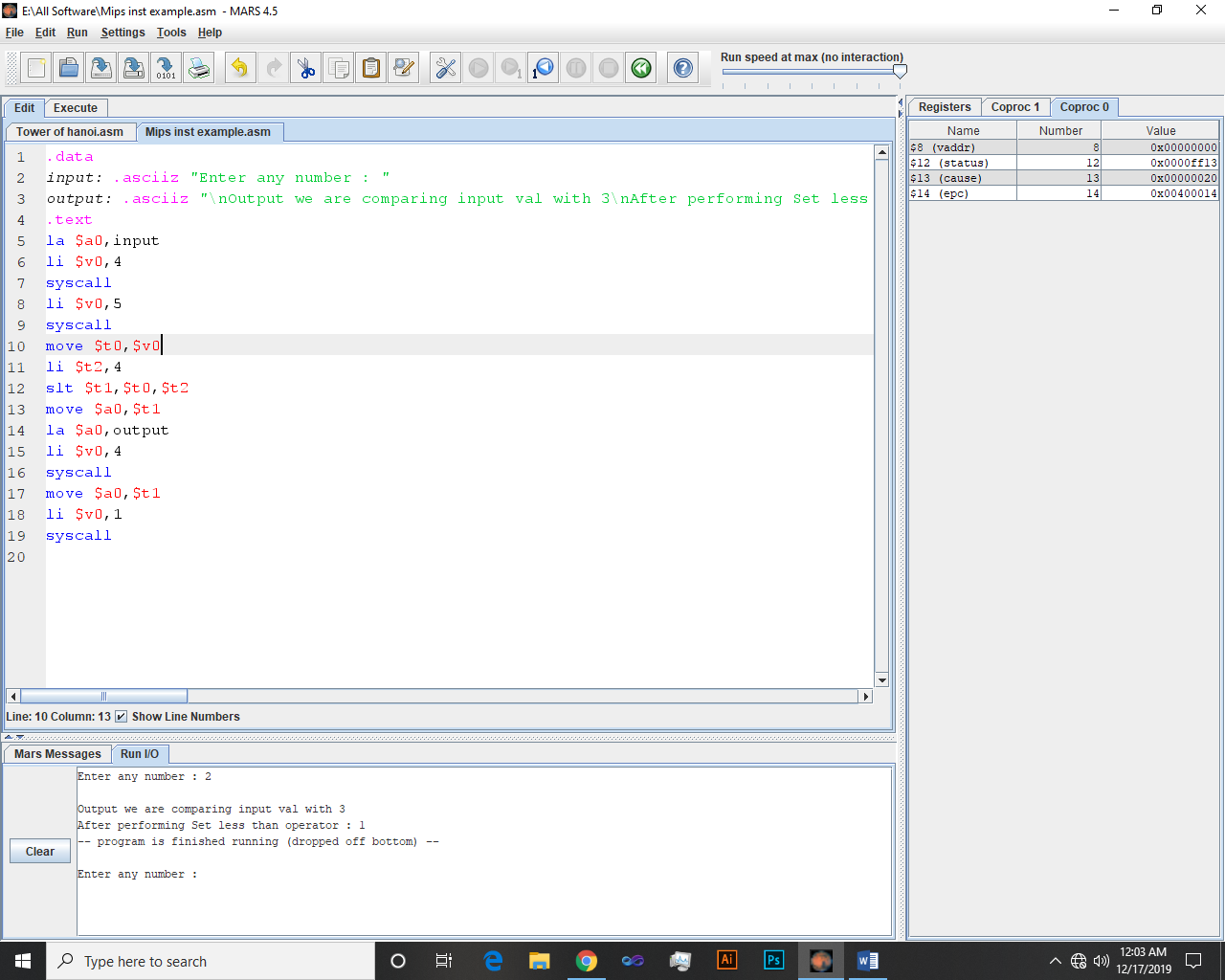
syscall

move $a0,$t1

li $v0,1

syscall

### Output:-



# sgt $t1,$t2,$t3

**Set Greater Than : if $t2 greater than $t3 then set $t1 to 1 else 0**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "\nOutput\nAfter performing Set greater than operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

li $t2,4

sgt $t1,$t0,$t2 #set greater than operator

move $a0,$t1

la $a0,output

li $v0,4

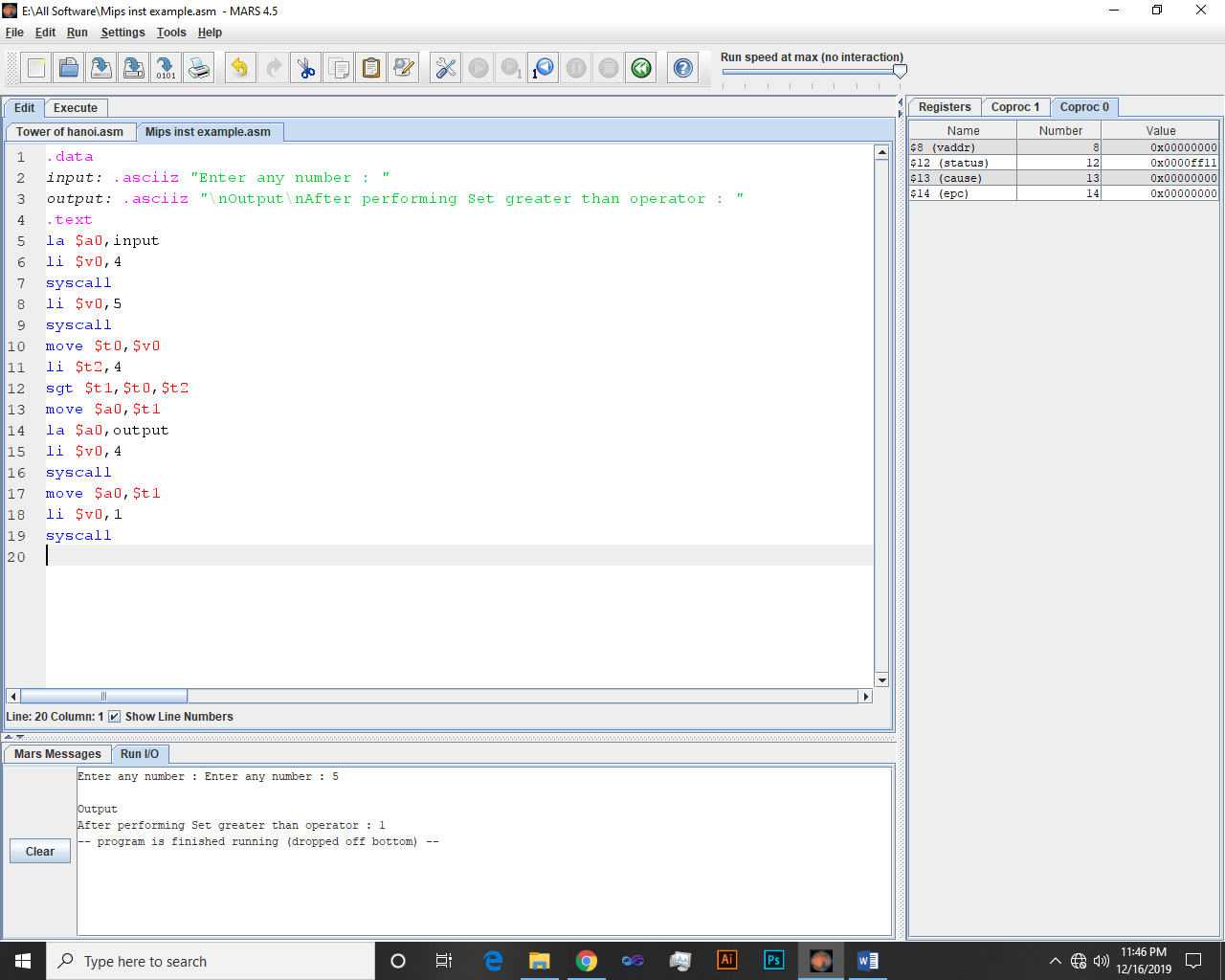
syscall

move $a0,$t1

li $v0,1

syscall

### Output:-



# sge $t1,$t2,$t3

**Set Greater or Equal : if $t2 greater or equal to $t3 then set $t1 to 1 else 0**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "\nOutput\nAfter performing Set greater or equal than operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

li $t2,4

sge $t1,$t0,$t2 #set greater than or equal

move $a0,$t1

la $a0,output

li $v0,4

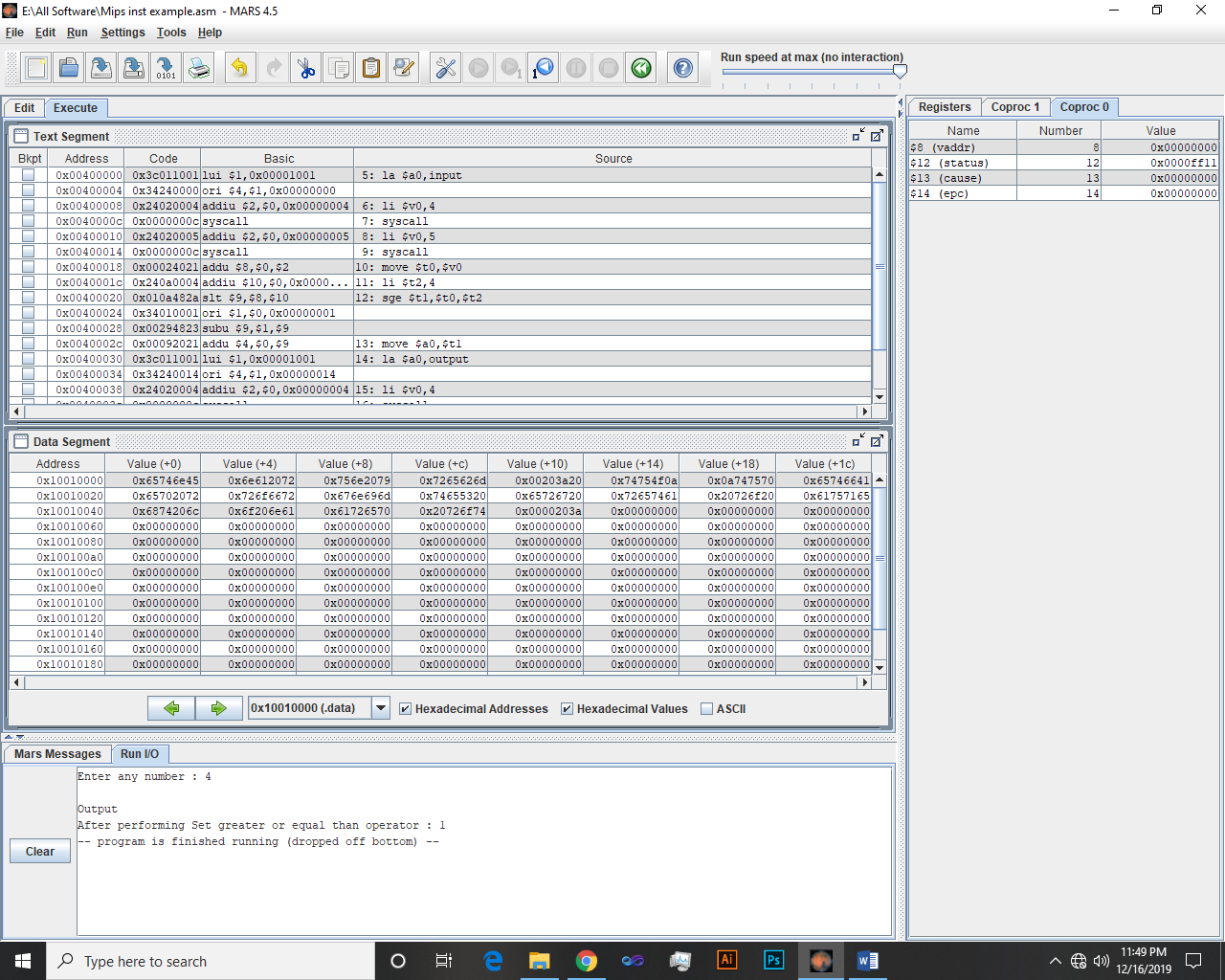
syscall

move $a0,$t1

li $v0,1

syscall

### Output:-



# seq $t1,$t2,$t3

**Set EQual : if $t2 equal to $t3 then set $t1 to 1 else 0**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "\nOutput\nAfter performing Set equal operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

li $t2,4

seq $t1,$t0,$t2 #set equal operator

move $a0,$t1

la $a0,output

li $v0,4

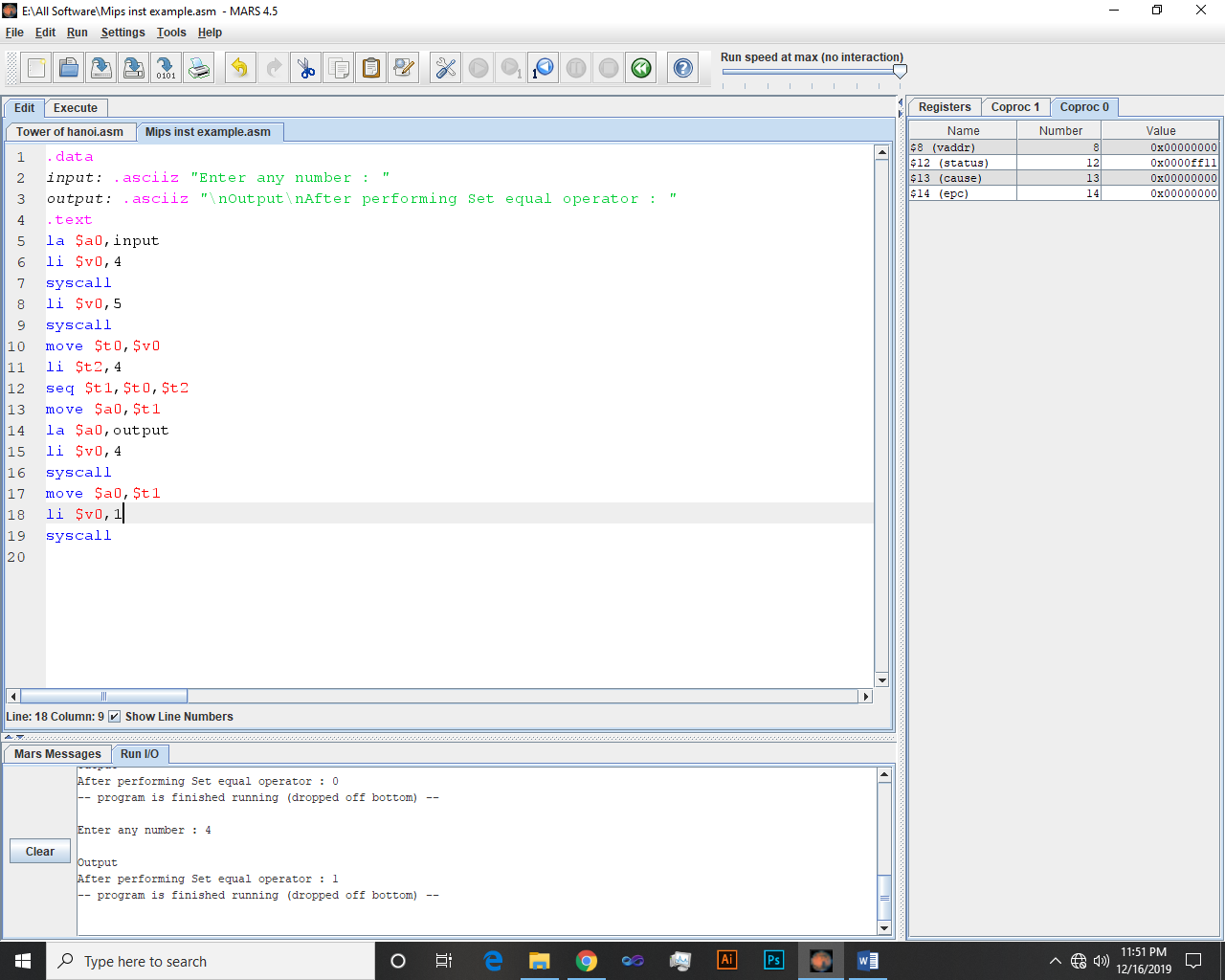
syscall

move $a0,$t1

li $v0,1

syscall

### Output:-



# abs.s $f0,$f1

**Floating point absolute value single precision : Set $f0 to absolute value of $f1, single precision**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "\nOutput\nAfter performing ABSolute value operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

abs $t1,$t0 #ABSolute value operator

move $a0,$t1

la $a0,output

li $v0,4

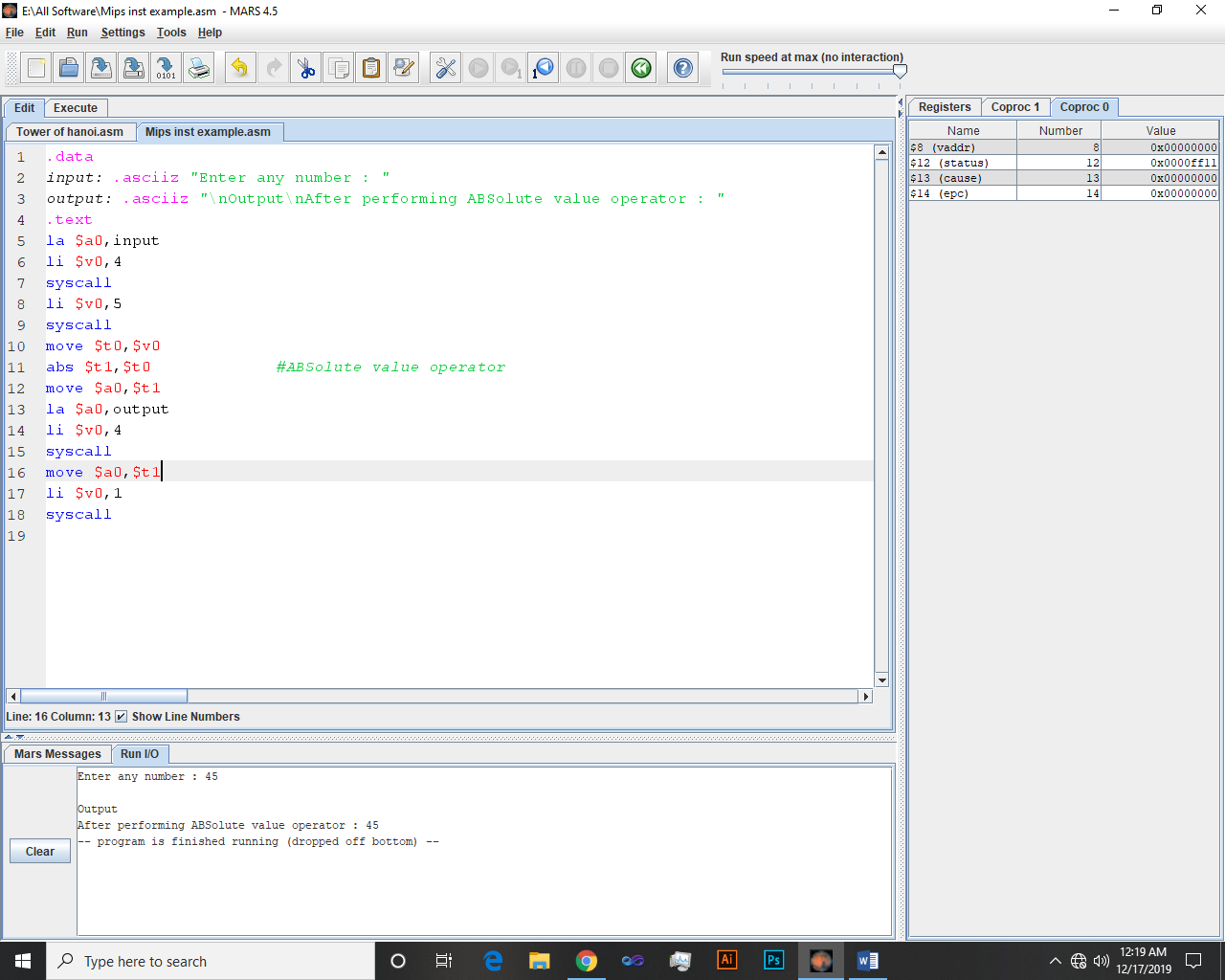
syscall

move $a0,$t1

li $v0,1

syscall

### Output:-



# nor $t1,$t2,$t3

**Bitwise NOR : Set $t1 to bitwise NOR of $t2 and $t3**

**Example:**

## Input:-

.data

output: .asciiz "\nOutput\nAfter performing NOR operator : "

.text

li $t0,1

li $t2,4

nor $t1,$t0,$t2 #NOR operator

move $a0,$t1

la $a0,output

li $v0,4

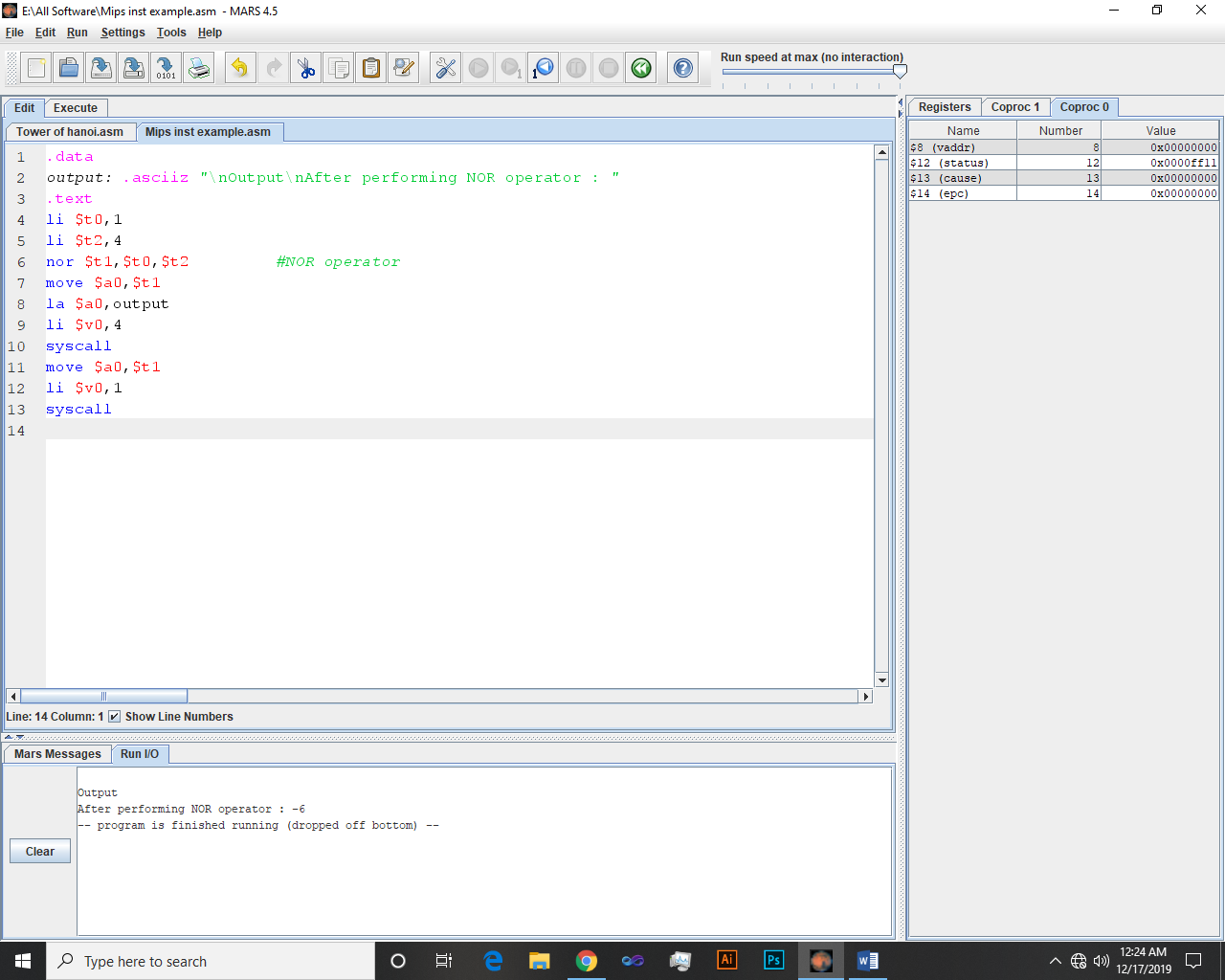
syscall

move $a0,$t1

li $v0,1

syscall

### Ouput:-



# not $t1,$t2

**Bitwise NOT (bit inversion)**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "\nOutput\nAfter performing NOT operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

not $t1,$t0 # not operator

move $a0,$t1

la $a0,output

li $v0,4

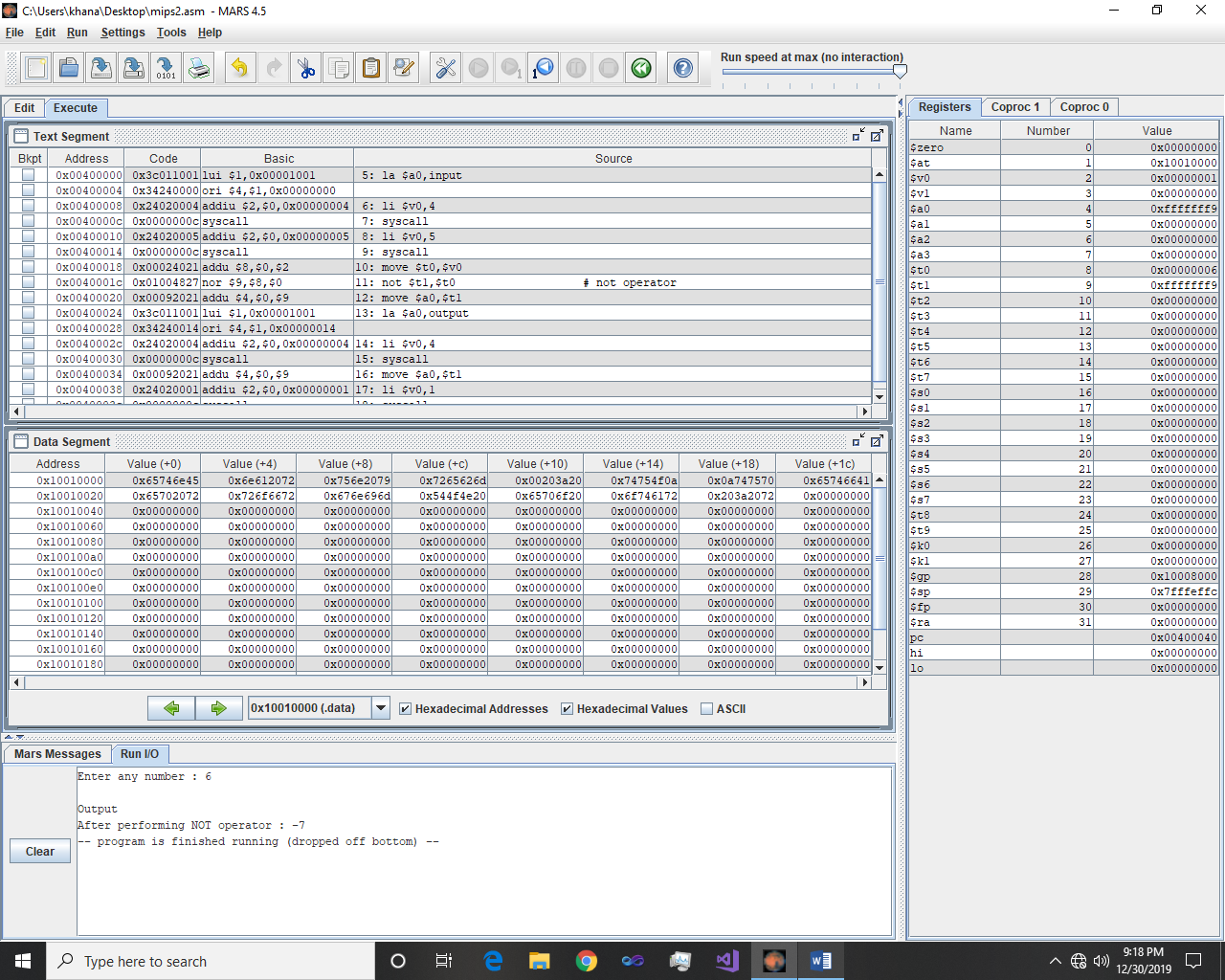
syscall

move $a0,$t1

li $v0,1

syscall

### Ouput:-



# rem $t1,$t2,$t3

**REMainder : Set $t1 to (remainder of $t2 divided by $t3)**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "\nOutput\nAfter performing Remainder operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

rem $t1,$t0,2 #remainder operator

move $a0,$t1

la $a0,output

li $v0,4

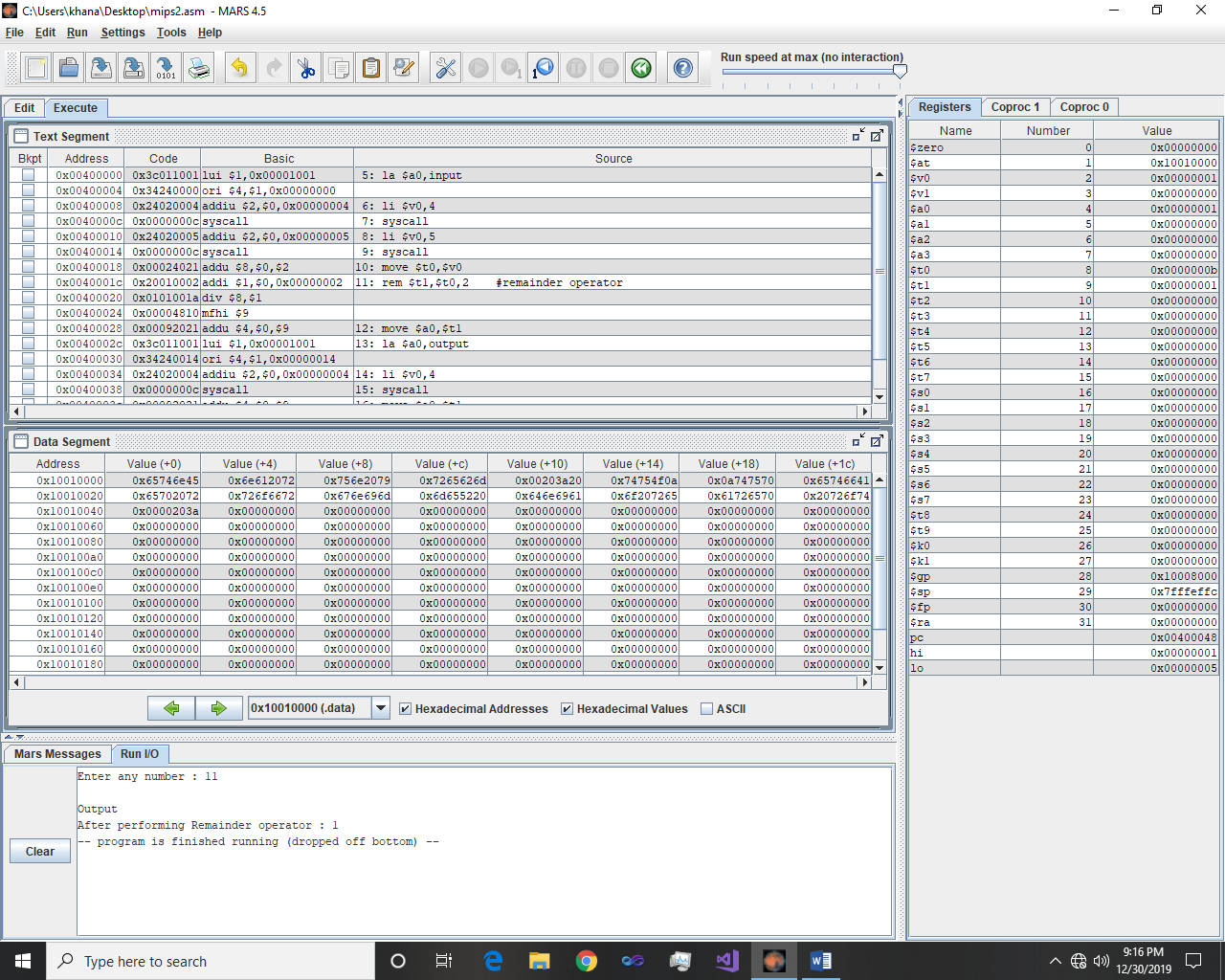
syscall

move $a0,$t1

li $v0,1

syscall

### Ouput:-



# ror $t1,$t2,$t3

**ROtate Right : Set $t1 to ($t2 rotated right by number of bit positions specified in $t3)**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "Output\nAfter performing Rotate right operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

li $t2,2

ror $t1,$t0,$t2 #Rotate right operator

la $a0,output

li $v0,4

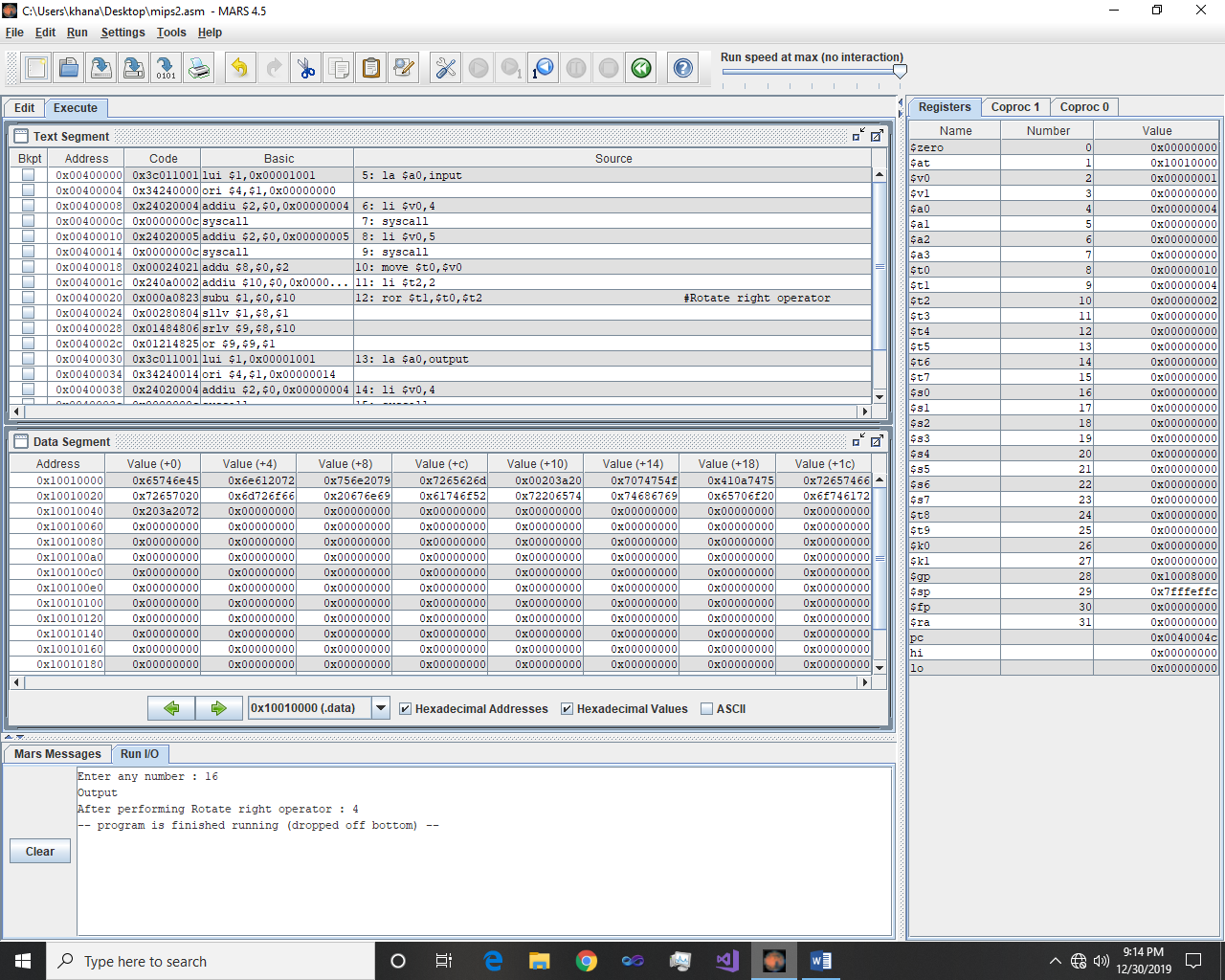
syscall

move $a0,$t1

li $v0,1

syscall

### Ouput:-



# rol $t1,$t2,$t3

**ROtate Left : Set $t1 to ($t2 rotated left by number of bit positions specified in $t3)**

**Example:**

## Input:-

.data

input: .asciiz "Enter any number : "

output: .asciiz "Output\nAfter performing Rotate left operator : "

.text

la $a0,input

li $v0,4

syscall

li $v0,5

syscall

move $t0,$v0

li $t2,2

rol $t1,$t0,$t2 #Rotate left operator

la $a0,output

li $v0,4

syscall

move $a0,$t1

li $v0,1

syscall

### Ouput:-

