# PRIMALITY CHECK IN MIPS

**LAB # 05**

**Fall 2023**

**CSE-304L Computer Organization and Architecture Lab**

Submitted by: **Ali Asghar**

Registration No.: **21PWCSE2059**

Class Section: **C**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Submitted to:

**Dr. Bilal Habib**

Date:

**2nd November 2023**

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

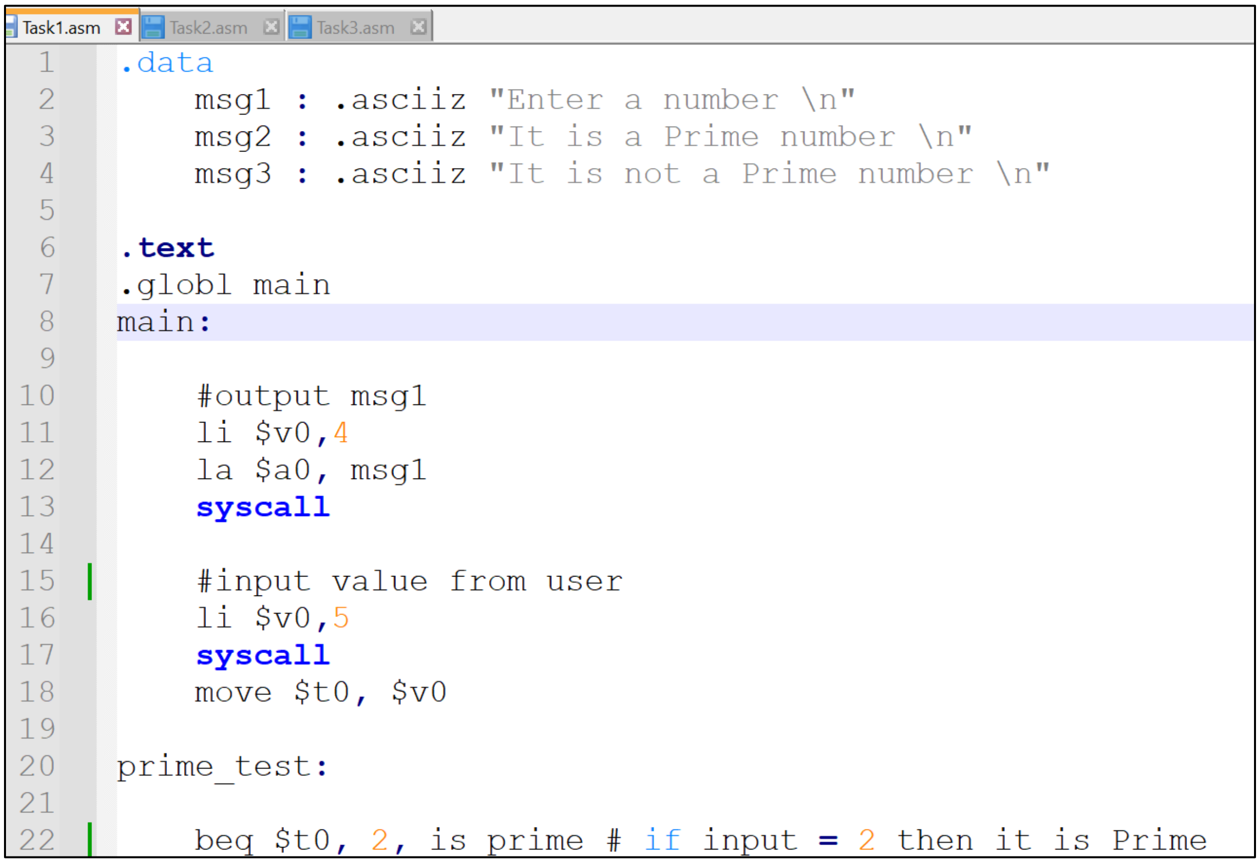
# ASSESSMENT RUBRICS COA LABS

| LAB REPORT ASSESSMENT | | | | |
|---|---|---|---|---|
| **Criteria** | **Excellent** | **Average** | **Nill** | **Marks Obtained** |
| **1. Objectives of Lab** | All objectives of lab are properly covered [Marks 10] | Objectives of lab are partially covered [Marks 5] | Objectives of lab are not shown [Marks 0] | |
| **2. MIPS instructions with Comments and proper indentations.** | All the instructions are well written with comments explaining the code and properly indented [Marks 20] | Some instructions are missing are poorly commented code [Marks 10] | The instructions are not properly written [Marks 0] | |
| **3. Simulation run without error and warnings** | The code is running in the simulator without any error and warnings [Marks 10] | The code is running but with some warnings or errors. [Marks 5] | The code is written but not running due to errors [Marks 0] | |
| **4. Procedure** | All the instructions are written with proper procedure [Marks 20] | Some steps are missing [Marks 10] | steps are totally missing [Marks 0] | |
| **5. OUTPUT** | Proper output of the code written in assembly [Marks 20] | Some of the outputs are missing [Marks 10] | No or wrong output [Marks 0] | |
| **6. Conclusion** | Conclusion about the lab is shown and written [Marks 20] | Conclusion about the lab is partially shown [Marks 10] | Conclusion about the lab is not shown[Marks0] | |
| **7. Cheating** | | | Any kind of cheating will lead to 0 Marks | |
| Total Marks Obtained:__________ Instructor Signature: ______________________ | | | | |

**Task 1:**

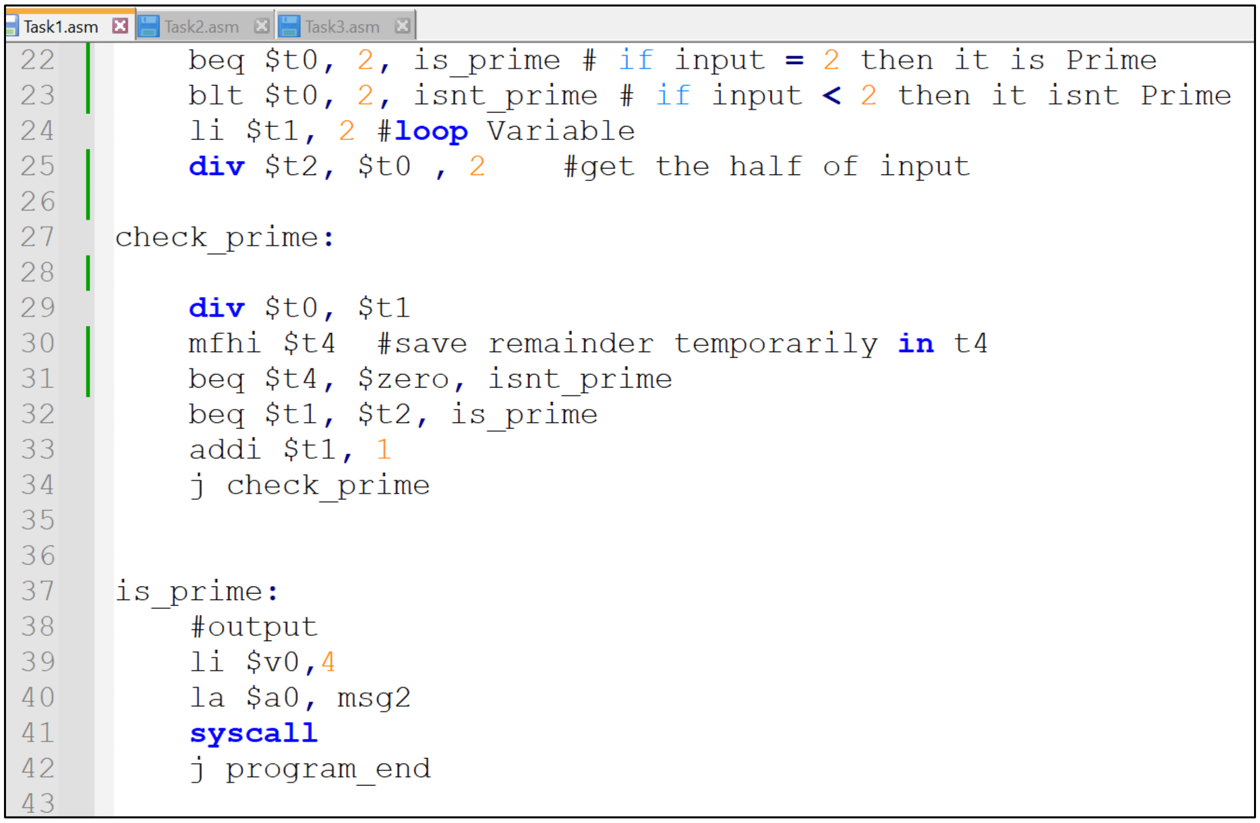
Write a program to check whether a number input by user is prime or not.

**Code:**

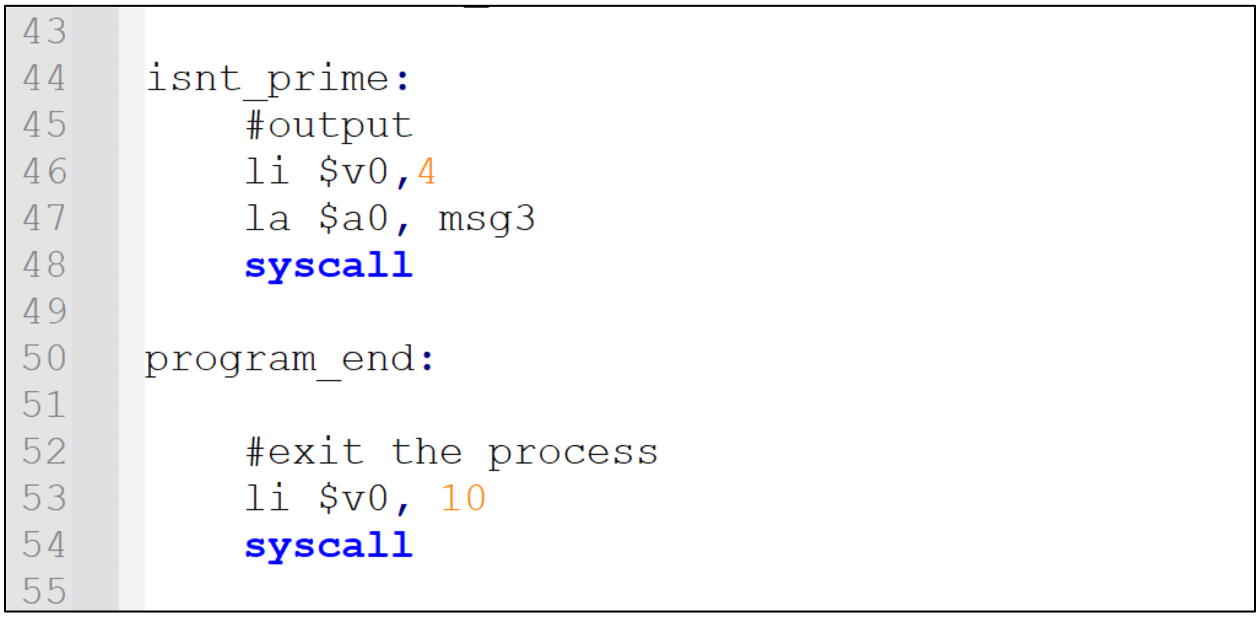


Task1.asm Task2.asm Task3.asm

```
.data
    msg1 : .asciiz "Enter a number \n"
    msg2 : .asciiz "It is a Prime number \n"
    msg3 : .asciiz "It is not a Prime number \n"

.text
.globl main
main:

    #output msg1
    li $v0,4
    la $a0, msg1
    syscall

    #input value from user
    li $v0,5
    syscall
    move $t0, $v0

prime_test:

    beq $t0, 2, is_prime # if input = 2 then it is Prime
```

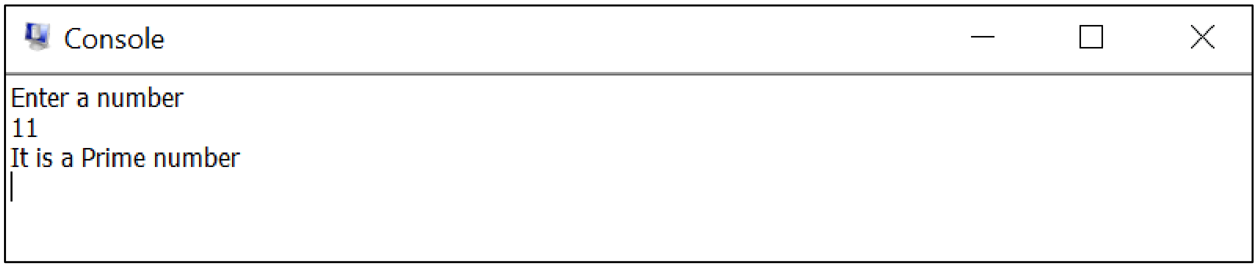


```
    beq $t0, 2, is_prime # if input = 2 then it is Prime
    blt $t0, 2, isnt_prime # if input < 2 then it isnt Prime
    li $t1, 2 #loop Variable
    div $t2, $t0 , 2    #get the half of input

check_prime:

    div $t0, $t1
    mfhi $t4  #save remainder temporarily in t4
    beq $t4, $zero, isnt_prime
    beq $t1, $t2, is_prime
    addi $t1, 1
    j check_prime


is_prime:
    #output
    li $v0,4
    la $a0, msg2
    syscall
    j program_end

```

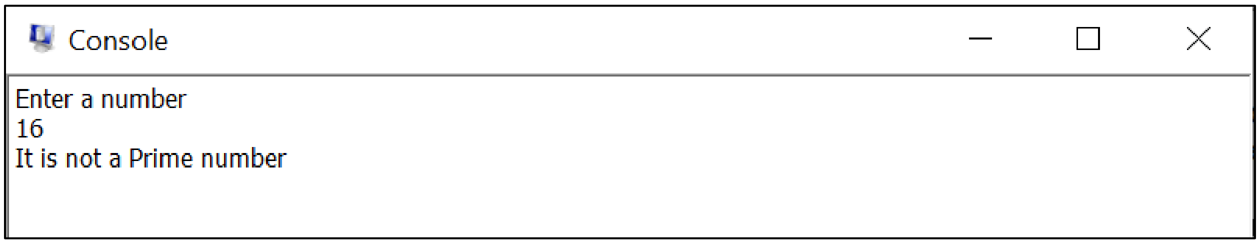


```

isnt_prime:
    #output
    li $v0,4
    la $a0, msg3
    syscall

program_end:

    #exit the process
    li $v0, 10
    syscall

```

**Output:**



```
Enter a number
11
It is a Prime number
```

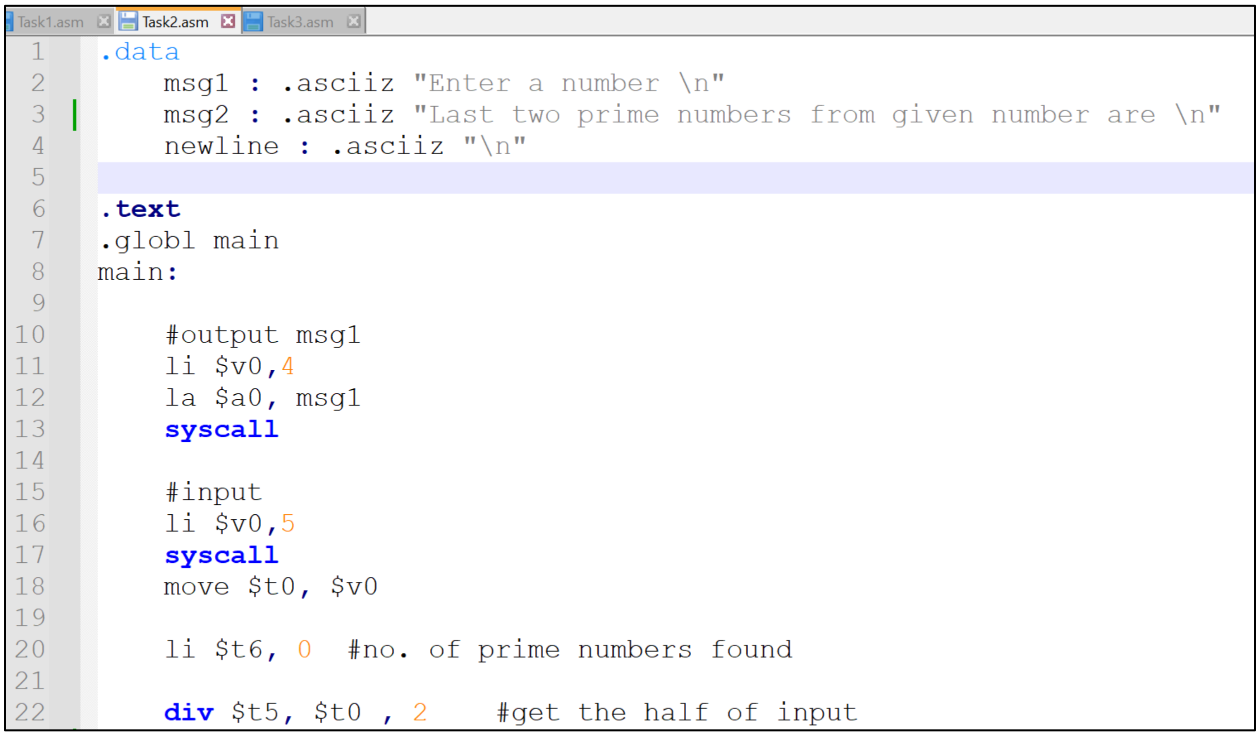


```
Enter a number
16
It is not a Prime number
```

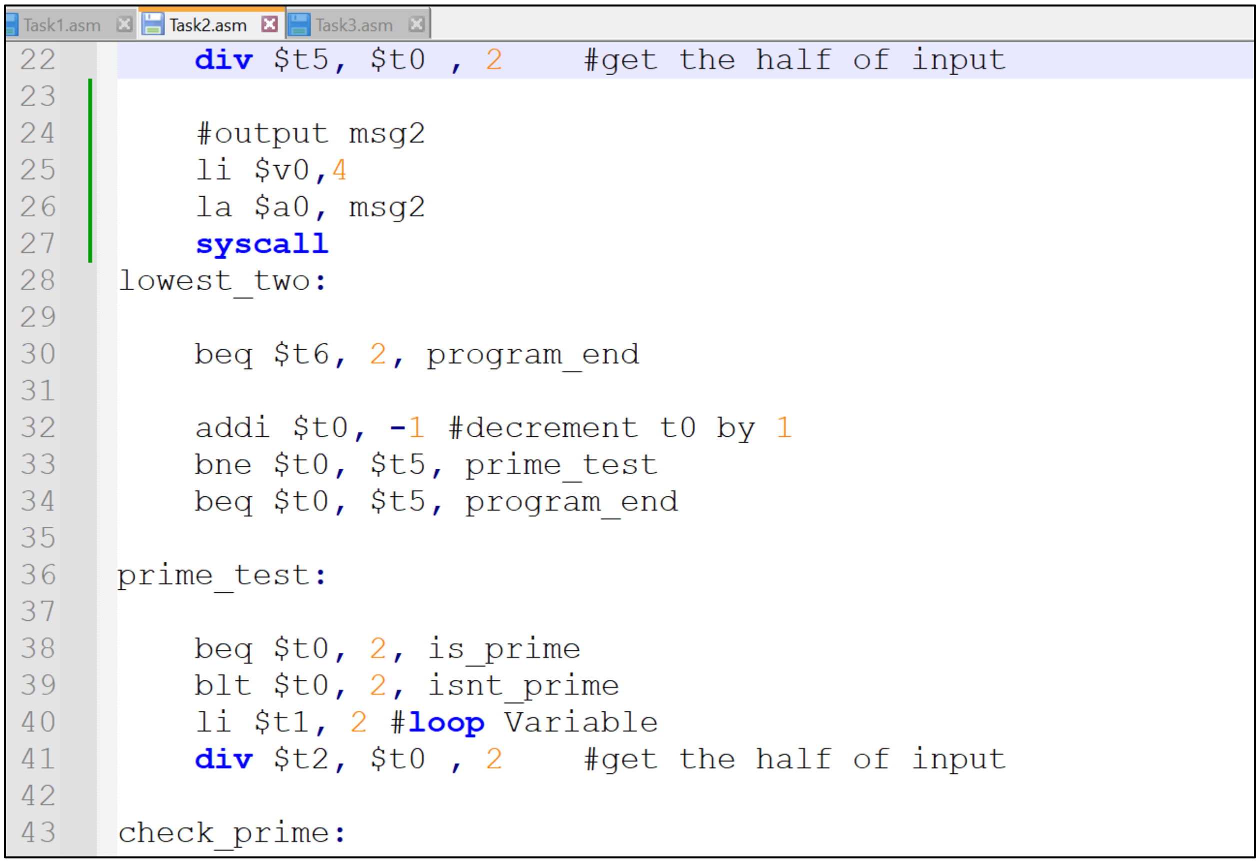
**Task 2:**

Repeat the above problem and display the largest two prime numbers lower than itself. Hint: If a user enters 20, then program displays 19 and 17.

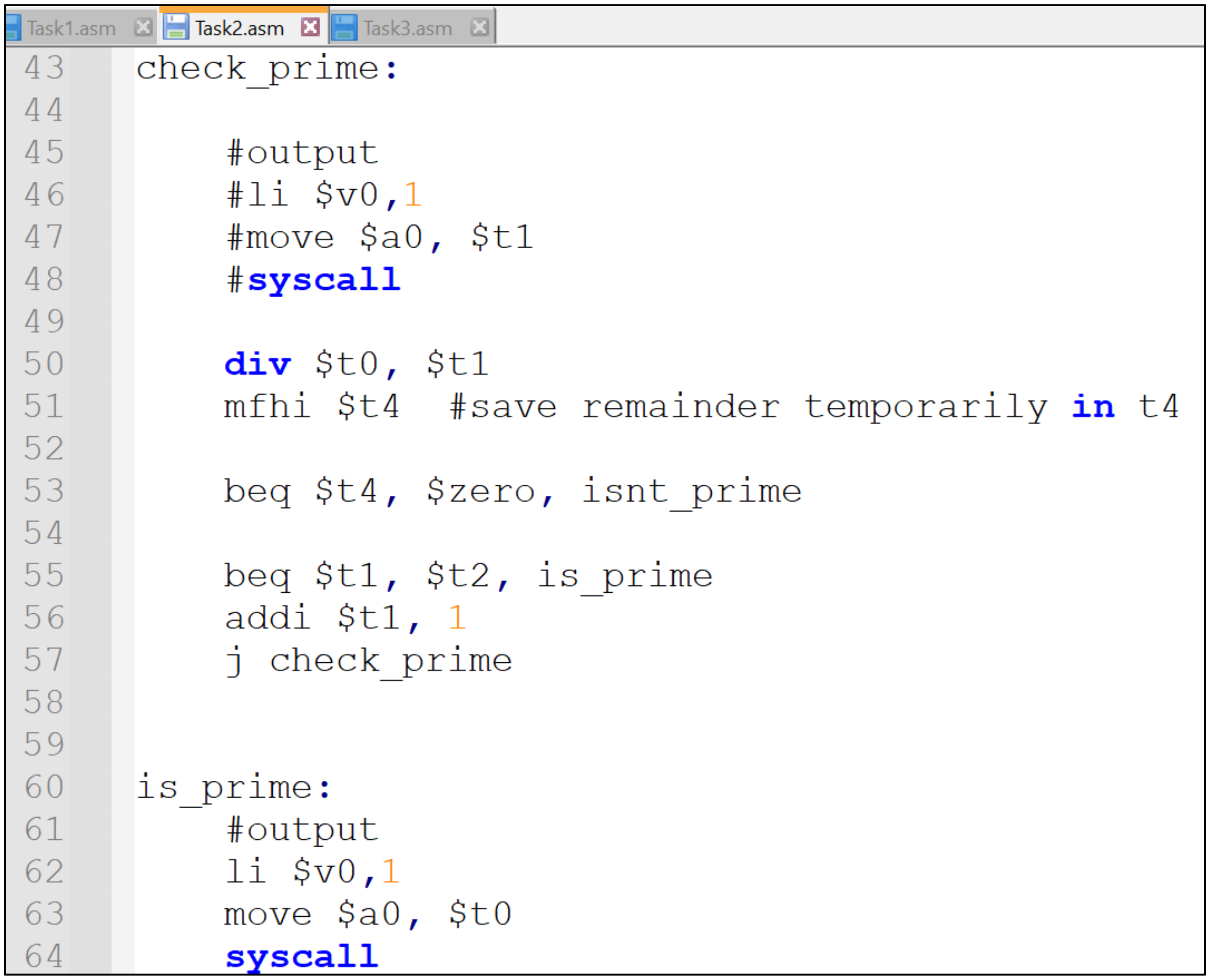
**Code:**



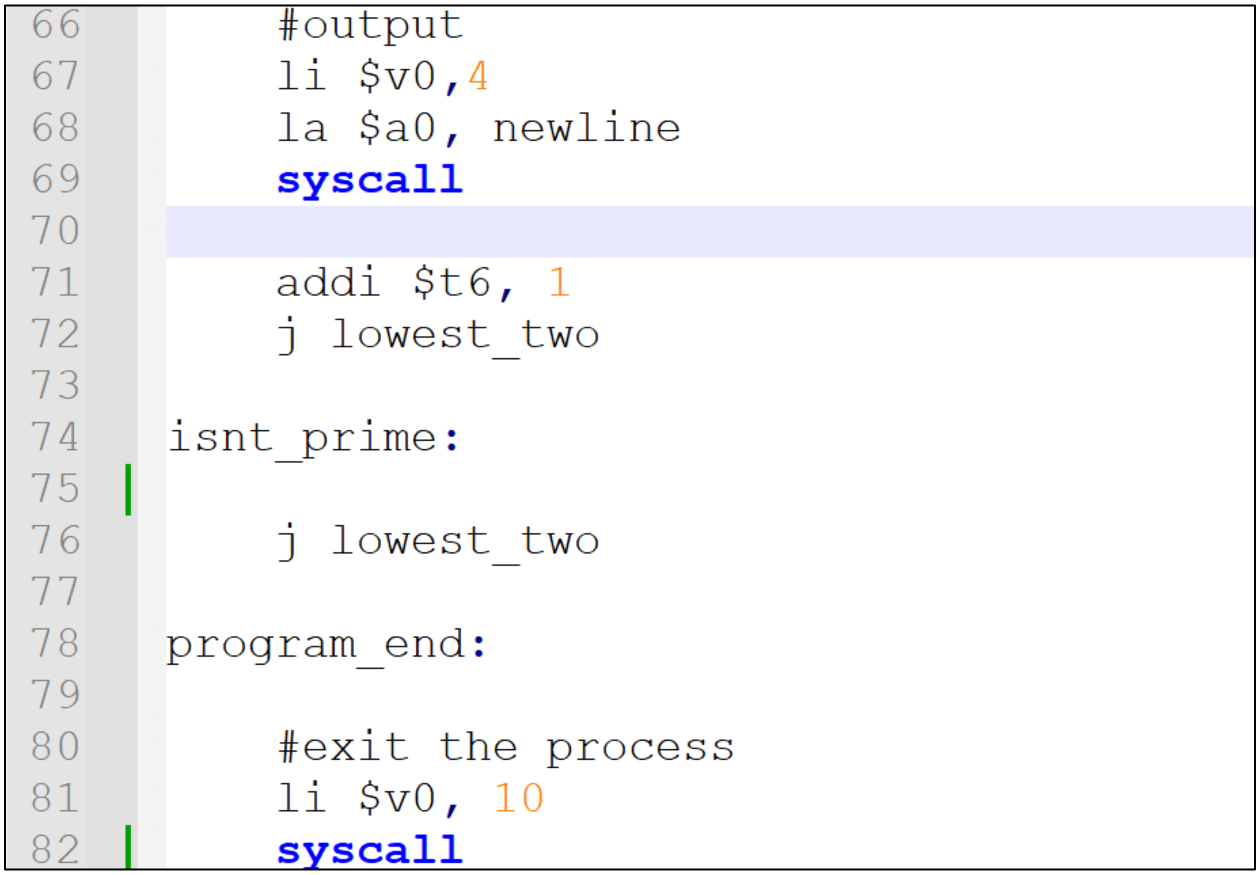
```
.data
    msg1 : .asciiz "Enter a number \n"
    msg2 : .asciiz "Last two prime numbers from given number are \n"
    newline : .asciiz "\n"

.text
.globl main
main:

    #output msg1
    li $v0,4
    la $a0, msg1
    syscall

    #input
    li $v0,5
    syscall
    move $t0, $v0

    li $t6, 0  #no. of prime numbers found

    div $t5, $t0 , 2    #get the half of input
```


Task1.asm Task2.asm Task3.asm

```
    div $t5, $t0 , 2     #get the half of input

    #output msg2
    li $v0,4
    la $a0, msg2
    syscall
lowest_two:

    beq $t6, 2, program_end

    addi $t0, -1 #decrement t0 by 1
    bne $t0, $t5, prime_test
    beq $t0, $t5, program_end

prime_test:

    beq $t0, 2, is_prime
    blt $t0, 2, isnt_prime
    li $t1, 2 #loop Variable
    div $t2, $t0 , 2     #get the half of input

check_prime:
```

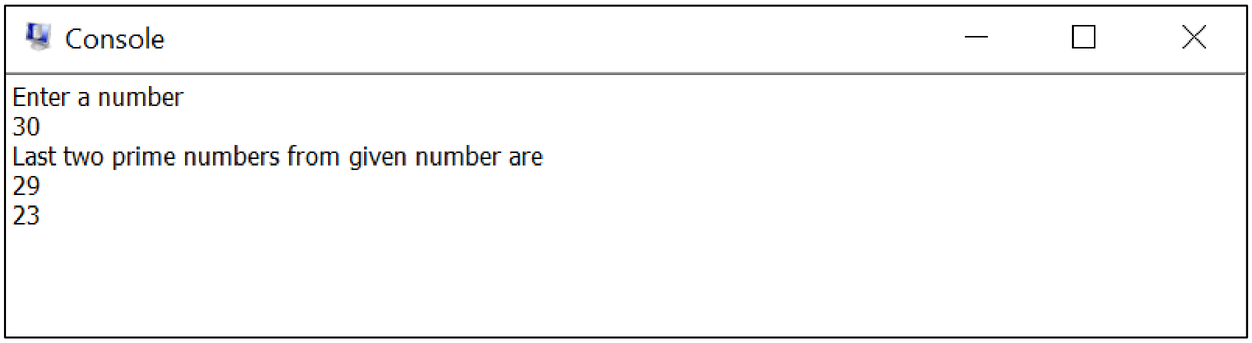

Task1.asm Task2.asm Task3.asm

```
check_prime:

    #output
    #li $v0,1
    #move $a0, $t1
    #syscall

    div $t0, $t1
    mfhi $t4  #save remainder temporarily in t4

    beq $t4, $zero, isnt_prime

    beq $t1, $t2, is_prime
    addi $t1, 1
    j check_prime


is_prime:
    #output
    li $v0,1
    move $a0, $t0
    syscall
```



```
    #output
    li $v0,4
    la $a0, newline
    syscall

    addi $t6, 1
    j lowest_two

isnt_prime:

    j lowest_two

program_end:

    #exit the process
    li $v0, 10
    syscall
```

**Output:**

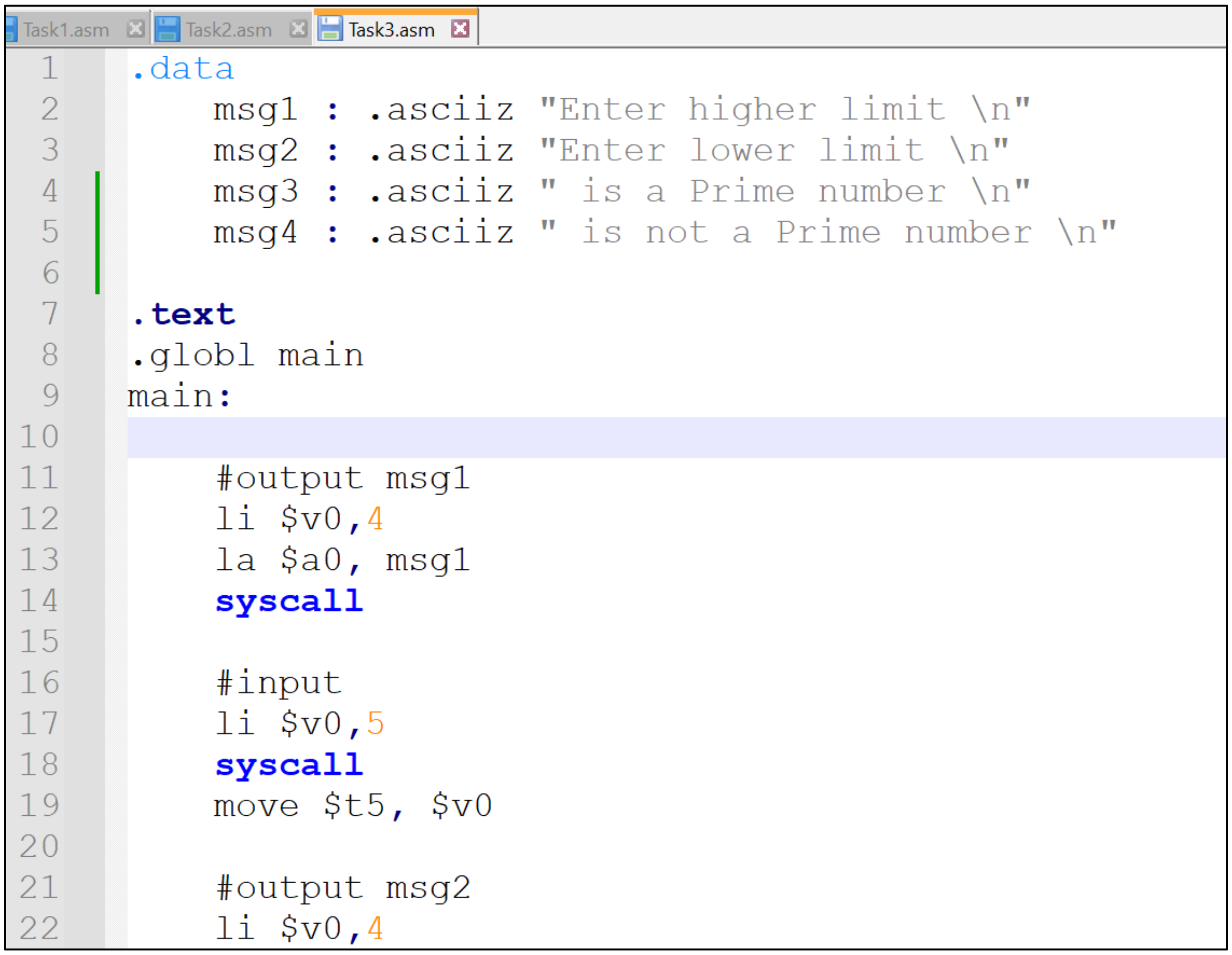


```
Enter a number
30
Last two prime numbers from given number are
29
23
```

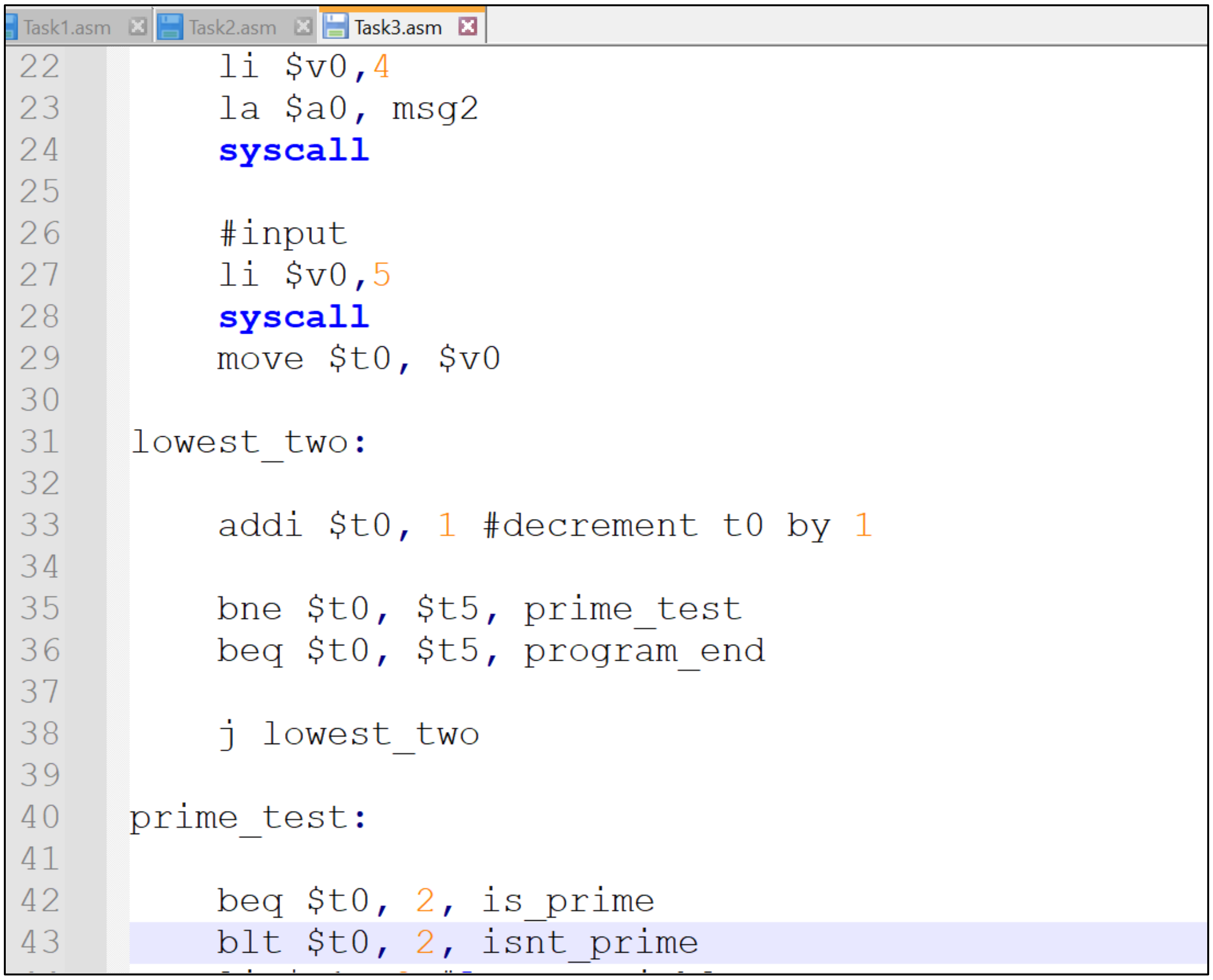
**Task 3:**

Write a program which takes two limits from user and display prime numbers between the two limits (if user enter lower limit 10 and upper limit 30 then display prime numbers between 10 and 30).

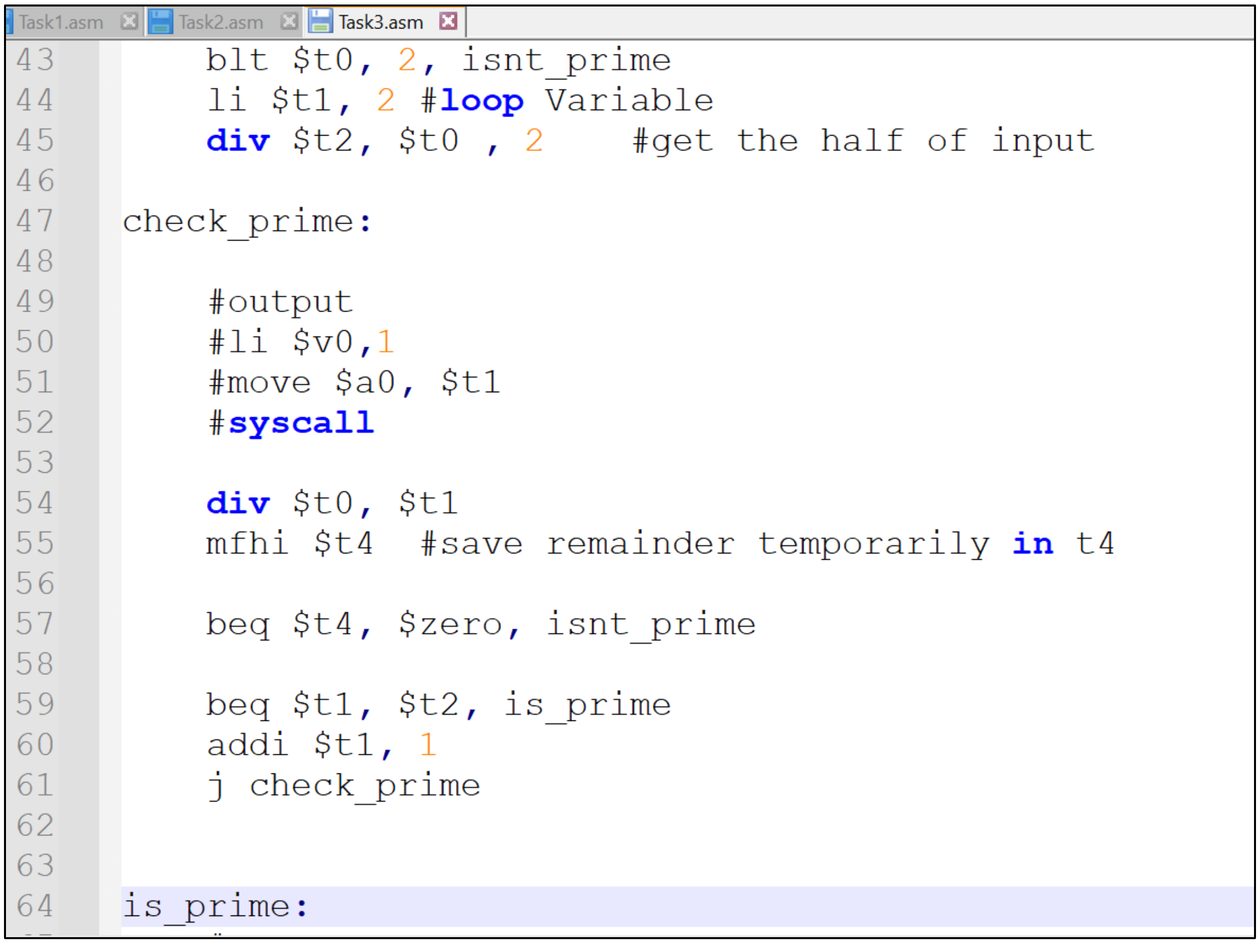
**Code:**



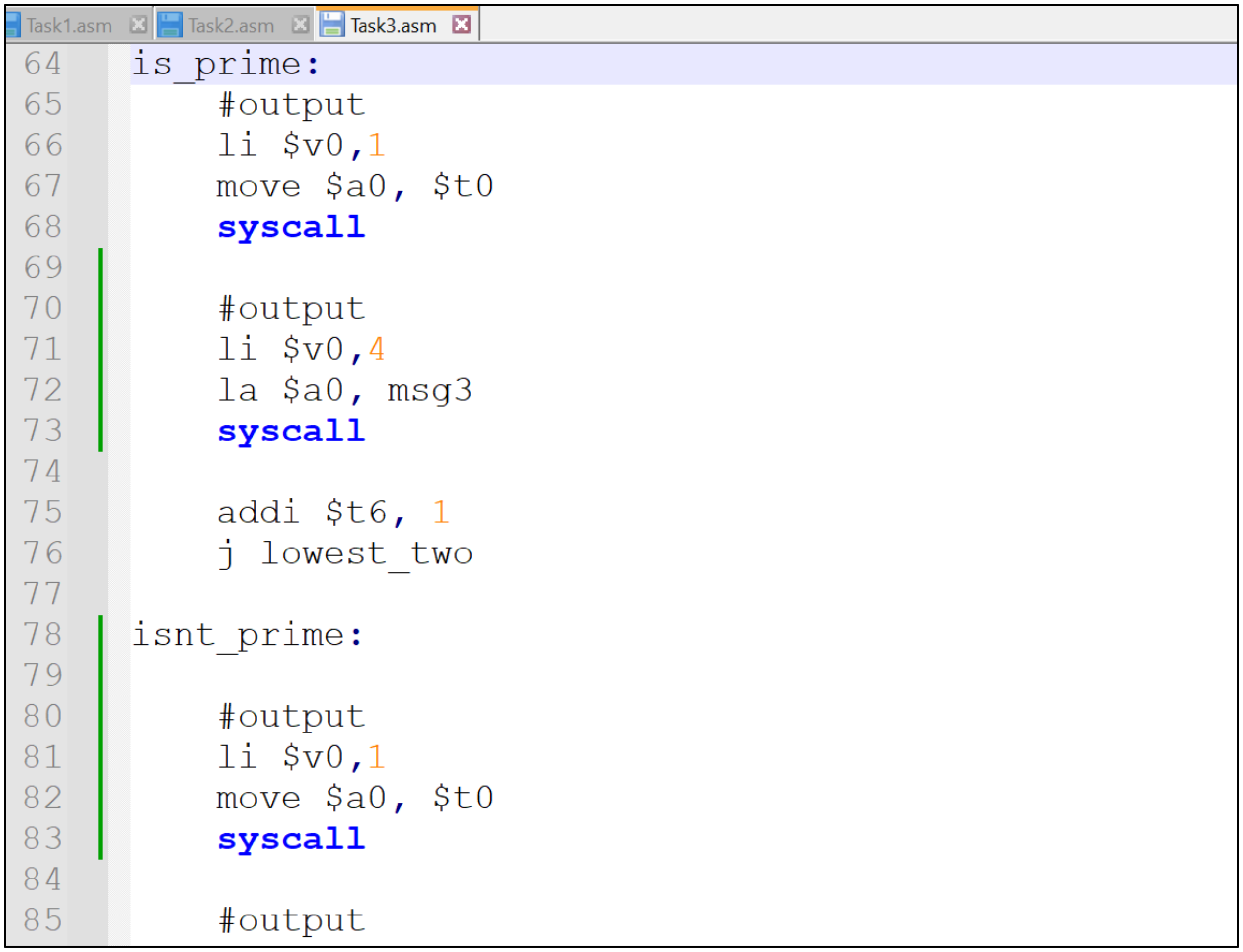
```
.data
    msg1 : .asciiz "Enter higher limit \n"
    msg2 : .asciiz "Enter lower limit \n"
    msg3 : .asciiz " is a Prime number \n"
    msg4 : .asciiz " is not a Prime number \n"

.text
.globl main
main:

    #output msg1
    li $v0,4
    la $a0, msg1
    syscall

    #input
    li $v0,5
    syscall
    move $t5, $v0

    #output msg2
    li $v0,4
```


Task1.asm Task2.asm Task3.asm

```
    li $v0,4
    la $a0, msg2
    syscall

    #input
    li $v0,5
    syscall
    move $t0, $v0

lowest_two:

    addi $t0, 1 #decrement t0 by 1

    bne $t0, $t5, prime_test
    beq $t0, $t5, program_end

    j lowest_two

prime_test:

    beq $t0, 2, is_prime
    blt $t0, 2, isnt_prime
```

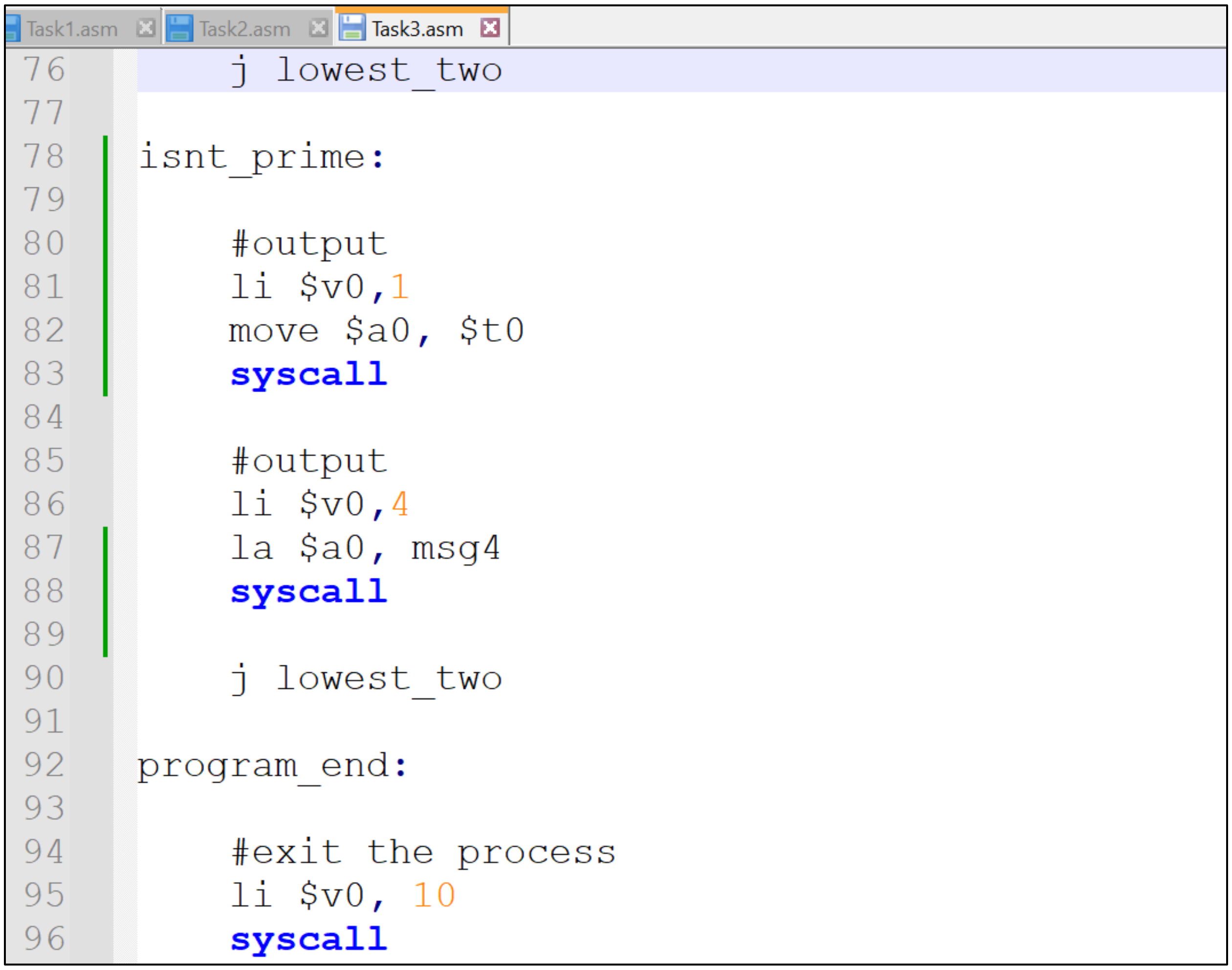


```
    blt $t0, 2, isnt_prime
    li $t1, 2 #loop Variable
    div $t2, $t0 , 2     #get the half of input

check_prime:

    #output
    #li $v0,1
    #move $a0, $t1
    #syscall

    div $t0, $t1
    mfhi $t4  #save remainder temporarily in t4

    beq $t4, $zero, isnt_prime

    beq $t1, $t2, is_prime
    addi $t1, 1
    j check_prime


is_prime:
```



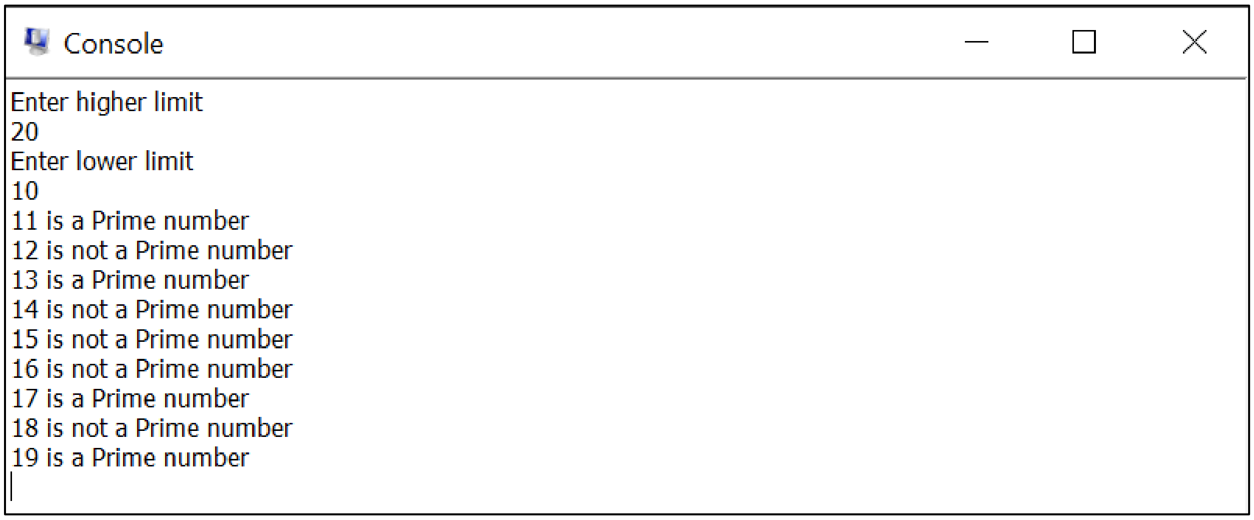
Task1.asm Task2.asm Task3.asm

```
is_prime:
    #output
    li $v0,1
    move $a0, $t0
    syscall

    #output
    li $v0,4
    la $a0, msg3
    syscall

    addi $t6, 1
    j lowest_two

isnt_prime:

    #output
    li $v0,1
    move $a0, $t0
    syscall

    #output
```



```
        j lowest_two

isnt_prime:

        #output
        li $v0,1
        move $a0, $t0
        syscall

        #output
        li $v0,4
        la $a0, msg4
        syscall

        j lowest_two

program_end:

        #exit the process
        li $v0, 10
        syscall
```

**Output:**



```
Enter higher limit
20
Enter lower limit
10
11 is a Prime number
12 is not a Prime number
13 is a Prime number
14 is not a Prime number
15 is not a Prime number
16 is not a Prime number
17 is a Prime number
18 is not a Prime number
19 is a Prime number
```

**Conclusion:**

In this lab, I learned about primality check (checking prime numbers) in MIPS Assembly.