

# **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:

**2<sup>nd</sup> November 2023**

**Department of Computer Systems Engineering**  
**University of Engineering and Technology, Peshawar**

---

**ASSESSMENT RUBRICS COA LABS**

---

<b>LAB REPORT ASSESSMENT</b>				
<b>Criteria</b>	<b>Excellent</b>	<b>Average</b>	<b>Nil</b>	<b>Marks Obtained</b>
<b>1. Objectives of Lab</b>	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]	
<b>2. MIPS instructions with Comments and proper indentations.</b>	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]	
<b>3. Simulation run without error and warnings</b>	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]	
<b>4. Procedure</b>	All the instructions are written with proper procedure [Marks 20]	Some steps are missing [Marks 10]	steps are totally missing [Marks 0]	
<b>5. OUTPUT</b>	Proper output of the code written in assembly [Marks 20]	Some of the outputs are missing [Marks 10]	No or wrong output [Marks 0]	
<b>6. Conclusion</b>	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]	
<b>7. Cheating</b>			Any kind of cheating will lead to 0 Marks	
<p style="text-align: right;">Total Marks Obtained: _____</p> <p style="text-align: right;">Instructor Signature: _____</p>				

### Task 1:

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

### Code:

```
Task1.asm Task2.asm Task3.asm
1  .data
2      msg1 : .asciiz "Enter a number \n"
3      msg2 : .asciiz "It is a Prime number \n"
4      msg3 : .asciiz "It is not a Prime number \n"
5
6  .text
7  .globl main
8  main:
9
10     #output msg1
11     li $v0,4
12     la $a0, msg1
13     syscall
14
15     #input value from user
16     li $v0,5
17     syscall
18     move $t0, $v0
19
20 prime_test:
21
22     beq $t0, 2, is_prime # if input = 2 then it is Prime
```

```
Task1.asm x Task2.asm x Task3.asm x
22      beq $t0, 2, is_prime # if input = 2 then it is Prime
23      blt $t0, 2, isnt_prime # if input < 2 then it isnt Prime
24      li $t1, 2 #loop Variable
25      div $t2, $t0, 2      #get the half of input
26
27      check_prime:
28
29          div $t0, $t1
30          mfhi $t4 #save remainder temporarily in t4
31          beq $t4, $zero, isnt_prime
32          beq $t1, $t2, is_prime
33          addi $t1, 1
34          j check_prime
35
36
37      is_prime:
38          #output
39          li $v0, 4
40          la $a0, msg2
41          syscall
42          j program_end
43
```

```
43
44      isnt_prime:
45          #output
46          li $v0, 4
47          la $a0, msg3
48          syscall
49
50      program_end:
51
52          #exit the process
53          li $v0, 10
54          syscall
55
```

## Output:

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

```
Console
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:

```
Task1.asm Task2.asm Task3.asm
1  .data
2      msg1 : .asciiz "Enter a number \n"
3      msg2 : .asciiz "Last two prime numbers from given number are \n"
4      newline : .asciiz "\n"
5
6  .text
7  .globl main
8  main:
9
10     #output msg1
11     li $v0,4
12     la $a0, msg1
13     syscall
14
15     #input
16     li $v0,5
17     syscall
18     move $t0, $v0
19
20     li $t6, 0 #no. of prime numbers found
21
22     div $t5, $t0 , 2 #get the half of input
```

```
Task1.asm x Task2.asm x Task3.asm x
22      div $t5, $t0 , 2      #get the half of input
23
24      #output msg2
25      li $v0,4
26      la $a0, msg2
27      syscall
28  lowest_two:
29
30      beq $t6, 2, program_end
31
32      addi $t0, -1 #decrement t0 by 1
33      bne $t0, $t5, prime_test
34      beq $t0, $t5, program_end
35
36  prime_test:
37
38      beq $t0, 2, is_prime
39      blt $t0, 2, isnt_prime
40      li $t1, 2 #loop Variable
41      div $t2, $t0 , 2      #get the half of input
42
43  check_prime:
```


```
Task1.asm x Task2.asm x Task3.asm x
43 check_prime:
44
45     #output
46     #li $v0,1
47     #move $a0, $t1
48     #syscall
49
50     div $t0, $t1
51     mfhi $t4 #save remainder temporarily in t4
52
53     beq $t4, $zero, isnt_prime
54
55     beq $t1, $t2, is_prime
56     addi $t1, 1
57     j check_prime
58
59
60 is_prime:
61     #output
62     li $v0,1
63     move $a0, $t0
64     syscall
```

```

66         #output
67         li $v0, 4
68         la $a0, newline
69         syscall
70
71         addi $t6, 1
72         j lowest_two
73
74     isnt_prime:
75
76         j lowest_two
77
78     program_end:
79
80         #exit the process
81         li $v0, 10
82         syscall

```

### Output:


Console

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:

```
Task1.asm x Task2.asm x Task3.asm x
1      .data
2      msg1 : .asciiz "Enter higher limit \n"
3      msg2 : .asciiz "Enter lower limit \n"
4      msg3 : .asciiz " is a Prime number \n"
5      msg4 : .asciiz " is not a Prime number \n"
6
7      .text
8      .globl main
9      main:
10
11      #output msg1
12      li $v0,4
13      la $a0, msg1
14      syscall
15
16      #input
17      li $v0,5
18      syscall
19      move $t5, $v0
20
21      #output msg2
22      li $v0,4
```

```
Task1.asm x Task2.asm x Task3.asm x
22      li $v0, 4
23      la $a0, msg2
24      syscall
25
26      #input
27      li $v0, 5
28      syscall
29      move $t0, $v0
30
31 lowest_two:
32
33      addi $t0, 1 #decrement t0 by 1
34
35      bne $t0, $t5, prime_test
36      beq $t0, $t5, program_end
37
38      j lowest_two
39
40 prime_test:
41
42      beq $t0, 2, is_prime
43      blt $t0, 2, isnt_prime
```

```
Task1.asm Task2.asm Task3.asm
43      blt $t0, 2, isnt_prime
44      li $t1, 2 #loop Variable
45      div $t2, $t0, 2 #get the half of input
46
47      check_prime:
48
49      #output
50      #li $v0, 1
51      #move $a0, $t1
52      #syscall
53
54      div $t0, $t1
55      mfhi $t4 #save remainder temporarily in t4
56
57      beq $t4, $zero, isnt_prime
58
59      beq $t1, $t2, is_prime
60      addi $t1, 1
61      j check_prime
62
63
64      is_prime:
```

```
Task1.asm x Task2.asm x Task3.asm x
64 is_prime:
65     #output
66     li $v0,1
67     move $a0, $t0
68     syscall
69
70     #output
71     li $v0,4
72     la $a0, msg3
73     syscall
74
75     addi $t6, 1
76     j lowest_two
77
78 isnt_prime:
79
80     #output
81     li $v0,1
82     move $a0, $t0
83     syscall
84
85     #output
```

```
Task1.asm x Task2.asm x Task3.asm x
76      j lowest_two
77
78      isnt_prime:
79
80      #output
81      li $v0,1
82      move $a0, $t0
83      syscall
84
85      #output
86      li $v0,4
87      la $a0, msg4
88      syscall
89
90      j lowest_two
91
92      program_end:
93
94      #exit the process
95      li $v0, 10
96      syscall
```

### Output:

```
Console
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.