Bahria University,

Karachi Campus

LAB EXPERIMENT NO.

07

LIST OF TASKS

TASK NO	OBJECTIVE
1.	Write a MIPS assembly program that reads an integer from the user and checks whether it is prime or not.
2.	Write a MIPS assembly program that prompts the user to enter three integers and then displays the largest among them.
3.	Write a MIPS assembly program that calculates the factorial of a number entered by the user.
4	Write a MIPS assembly program that takes an integer input from the user and displays the sum of all even numbers up to that number.
5	Write a MIPS assembly program that calculates the perimeter of a shape (circle, square, rectangle, triangle, trapezium, or pentagon) based on the user's choice.

Submitted On:

Date: 21/10/2025

Task No. 01:

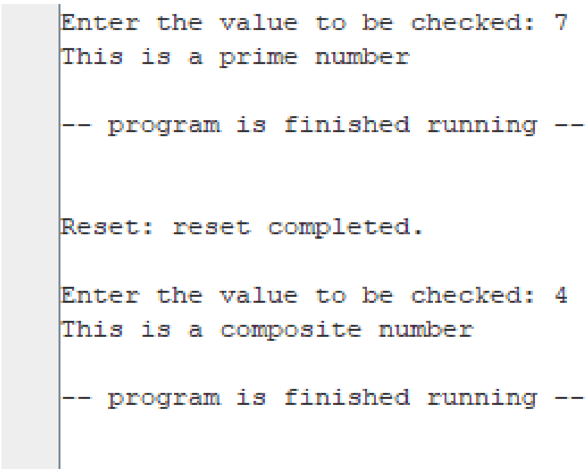
"Write a MIPS assembly program that reads an integer from the user and checks whether it is prime or not."

Solution:

```
.data
inputMsg: .asciiz "Enter the value to be checked: "
newLine: .asciiz "\n"
primeMsg: .asciiz "This is a prime number\n"
compositeMsg: .asciiz "This is a composite number\n"

.text
.globl main
main:
   li $v0,4
   la $a0,inputMsg
   syscall

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

   ble $t0, 1,notPrime

   li $t1, 2

loop:
   mul $t3,$t1,$t1
   bgt $t3,$t0,prime

   div $t0,$t1
   mfhi $t4
   beqz $t4,notPrime

   addi $t1, $t1,1
   b loop

notPrime:
   li $v0, 4
   la $a0,compositeMsg
   syscall
   b exit

prime:
   li $v0,4
```

```
    la $a0,primeMsg
    syscall
    b exit

exit:
    li $v0, 10
    syscall
```

Output:



```
Enter the value to be checked: 7
This is a prime number

-- program is finished running --


Reset: reset completed.

Enter the value to be checked: 4
This is a composite number

-- program is finished running --
```

Task No. 02:

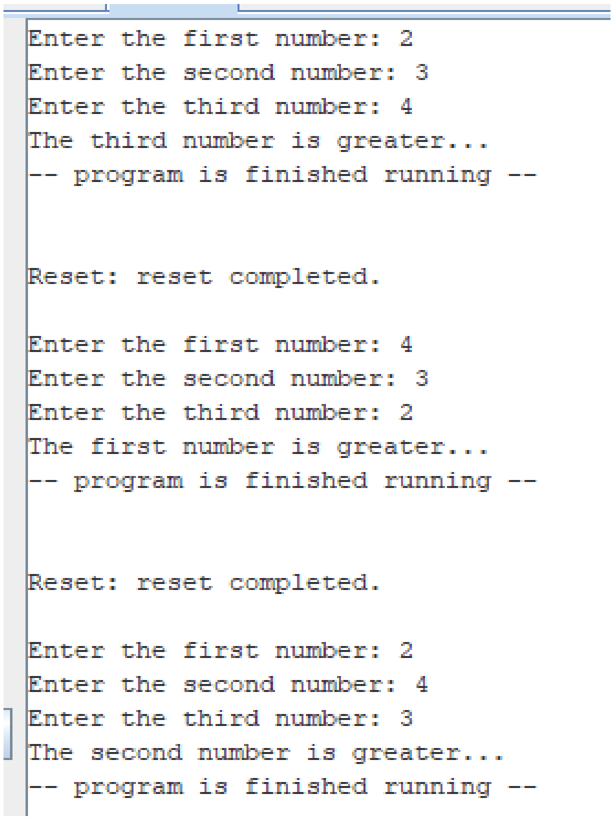
“Write a MIPS assembly program that prompts the user to enter three integers and then displays the largest among them.”

Solution:

```
.data
input1: .asciiz "Enter the first number: "
input2: .asciiz "Enter the second number: "
input3: .asciiz "Enter the third number: "
greater1: .asciiz "The first number is greater..."
greater2: .asciiz "The second number is greater..."
greater3: .asciiz "The third number is greater..."
newLine: .asciiz "\n"

.text
.globl main
```

```
main:
    li $v0,4
    la $a0,input1
    syscall
    li $v0,5
    syscall
    move $t1,$v0

    li $v0,4
    la $a0,input2
    syscall
    li $v0,5
    syscall
    move $t2,$v0

    li $v0,4
    la $a0,input3
    syscall
    li $v0,5
    syscall
    move $t3,$v0

    bgt $t1,$t2,check13
    b check2

check13:
    bgt $t1,$t3,oneGreater
    b check2

check2:
    bgt $t2,$t1,check23
    b check3

check23:
    bgt $t2,$t3,twoGreater
    b check3

check3:
    bgt $t3,$t1,check32
    b oneGreater

check32:
    bgt $t3,$t2threeGreater
    b oneGreater

oneGreater:
    li $v0,4
    la $a0,greater1
    syscall
    b exit

twoGreater:
    li $v0,4
    la $a0,greater2
    syscall
    b exit

threeGreater:
    li $v0,4
    la $a0,greater3
    syscall
    b exit

exit:
    li $v0,10
    syscall
```

Output:



```
Enter the first number: 2
Enter the second number: 3
Enter the third number: 4
The third number is greater...
-- program is finished running --


Reset: reset completed.

Enter the first number: 4
Enter the second number: 3
Enter the third number: 2
The first number is greater...
-- program is finished running --


Reset: reset completed.

Enter the first number: 2
Enter the second number: 4
Enter the third number: 3
The second number is greater...
-- program is finished running --
```

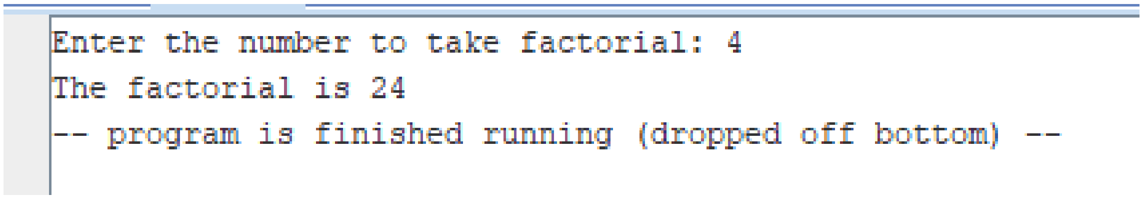
Task No. 03:

“Write a MIPS assembly program that calculates the factorial of a number entered by the user”.

Solution:

```
.data
inputMsg: .asciiz "Enter the number to take factorial: "
factorial: .asciiz "The factorial is "

.text
.globl main
main:
   li $v0,4
   la $a0,inputMsg
   syscall
   li $v0,5
   syscall
   move $t0,$v0
   li $t1,1

while:
   beqz  $t0,exit
   b multi
multi:
   mul $t1,$t1,$t0
   subi $t0,$t0,1
   b while

exit:
   li $v0,4
   la $a0,factorial
   syscall

   li $v0,1
   move $a0,$t1
   syscall
```

Output:



```
Enter the number to take factorial: 4
The factorial is 24
-- program is finished running (dropped off bottom) --
```

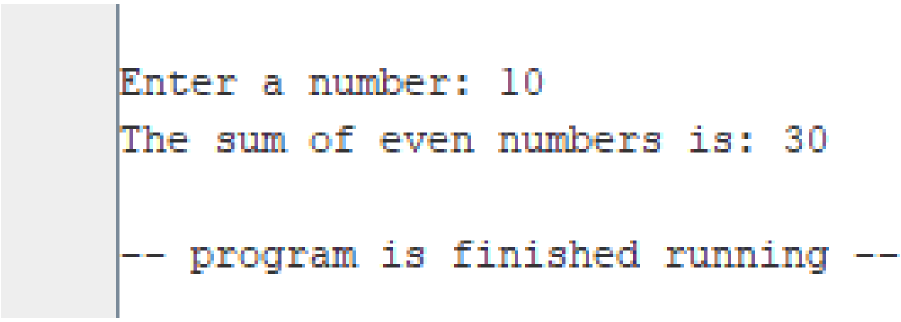
Task No. 04:

“Write a MIPS assembly program that takes an integer input from the user and displays the sum of all even numbers up to that number”.

Solution:

```
.data
inputMsg: .asciiz "Enter a number: "
outputMsg: .asciiz "The sum of even numbers is: "
newLine:  .asciiz "\n"

.text
.globl main
main:
    li $v0,4
    la $a0,inputMsg
    syscall

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

    li $t1,2
    li $t2,0

loop:
    bgt $t1,$t0,printing
    add $t2,$t2,$t1
    addi $t1,$t1,2
    b loop

printing:
    li $v0,4
    la $a0,outputMsg
    syscall

    li $v0, 1
    move $a0,$t2
    syscall

    li $v0,4
    la $a0,newLine
    syscall

    li $v0,10
    syscall
```

Output:



```
Enter a number: 10
The sum of even numbers is: 30

-- program is finished running --
```

Task No. 05:

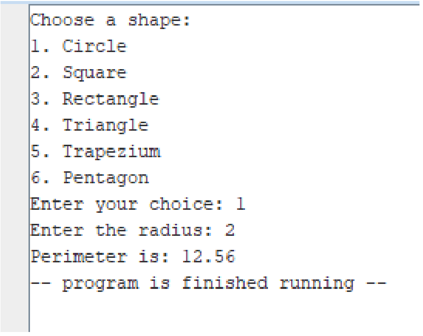
“Write a MIPS assembly program that calculates the perimeter of a shape (circle, square, rectangle, triangle, trapezium, or pentagon) based on the user’s choice.”

Solution:

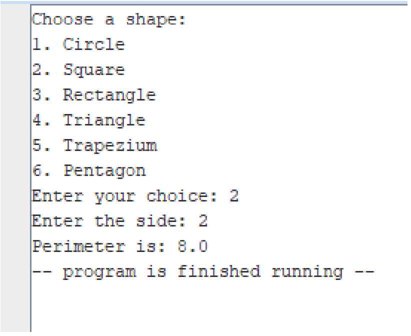
```
.data
menu: .asciiz "Choose a shape:\n1. Circle\n2. Square\n3. Rectangle\n4. Triangle\n5. Trapezium\n6. Pentagon\nEnter your choice: "
radiusMsg: .asciiz "Enter the radius: "
sideMsg: .asciiz "Enter the side: "
lengthMsg: .asciiz "Enter the length: "
widthMsg: .asciiz "Enter the width: "
aMsg: .asciiz "Enter side a: "
bMsg: .asciiz "Enter side b: "
cMsg: .asciiz "Enter side c: "
dMsg: .asciiz "Enter side d: "
perimeterMsg: .asciiz "Perimeter is: "
pi: .float 3.14
two: .float 2.0
four: .float 4.0
five: .float 5.0
newLine: .asciiz "\n"

.text
.globl main
main:
   li $v0,4
   la $a0,menu
   syscall

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

   beq $t0,1,circle
   beq $t0,2,square
   beq $t0,3,rectangle
   beq $t0,4,triangle
   beq $t0,5,trapezium
   beq $t0,6,pentagon
   b exit

circle:
   li $v0,4
   la $a0,radiusMsg
   syscall

   li $v0,6
   syscall
   mov.s $f1,$f0

   l.s $f2,pi
   l.s $f3,two
   mul.s $f4,$f2,$f3
   mul.s $f12,$f4,$f1

   li $v0,4
   la $a0,perimeterMsg
   syscall

   li $v0,2
   syscall
   b exit

square:
   li $v0,4
   la $a0,sideMsg
   syscall

   li $v0,6
   syscall
   mov.s $f1,$f0

   l.s $f2,four
   mul.s $f12,$f1,$f2

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

```
    li $v0,2
    syscall
    b exit

 rectangle:
    li $v0,4
    la $a0,lengthMsg
    syscall

    li $v0,6
    syscall
    mov.s $f1,$f0

    li $v0,4
    la $a0,widthMsg
    syscall

    li $v0,6
    syscall
    mov.s $f2,$f0

    add.s $f3,$f1,$f2
    l.s $f4,two
    mul.s $f12,$f3,$f4

    li $v0,4
    la $a0,perimeterMsg
    syscall

    li $v0,2
    syscall
    b exit

 triangle:
    li $v0,4
    la $a0,aMsg
    syscall
    li $v0,6
    syscall
    mov.s $f1,$f0

    li $v0,4
    la $a0,bMsg
    syscall
    li $v0,6
    syscall
    mov.s $f2,$f0

    li $v0,4
    la $a0,cMsg
    syscall
    li $v0,6
    syscall
    mov.s $f3,$f0

    add.s $f12,$f1,$f2
    add.s $f12,$f12,$f3

    li $v0,4
    la $a0,perimeterMsg
    syscall

    li $v0,2
    syscall
    b exit

 trapezium:
    li $v0,4
    la $a0,aMsg
    syscall
    li $v0,6
    syscall
    mov.s $f1,$f0

    li $v0,4
    la $a0,bMsg
    syscall
    li $v0,6
    syscall
    mov.s $f2,$f0

    li $v0,4
    la $a0,cMsg
    syscall
    li $v0,6
    syscall
    mov.s $f3,$f0

    li $v0,4
    la $a0,dMsg
    syscall
    li $v0,6
    syscall
    mov.s $f4,$f0

    add.s $f12,$f1,$f2
    add.s $f12,$f12,$f3
    add.s $f12,$f12,$f4
    li $v0,4
    la $a0,perimeterMsg
    syscall

    li $v0,2
    syscall
    b exit

pentagon:
    li $v0,4
    la $a0,sideMsg
    syscall

    li $v0,6
    syscall
    mov.s $f1,$f0

    l.s $f2,five
    mul.s $f12,$f1,$f2

    li $v0,4
    la $a0,perimeterMsg
    syscall

    li $v0,2
    syscall
    b exit

exit:
    li $v0,10
    syscall
```

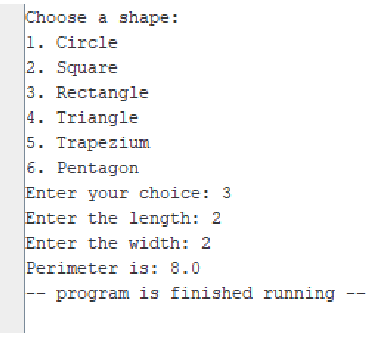
Outputs:



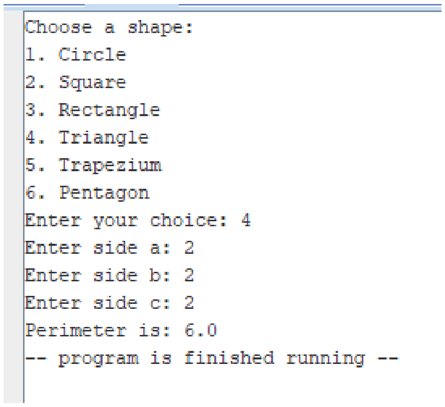
```
Choose a shape:
1. Circle
2. Square
3. Rectangle
4. Triangle
5. Trapezium
6. Pentagon
Enter your choice: 1
Enter the radius: 2
Perimeter is: 12.56
-- program is finished running --
```



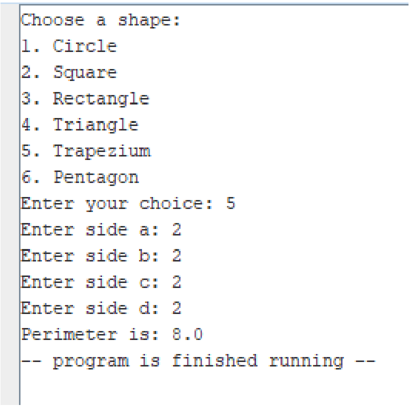
```
Choose a shape:
1. Circle
2. Square
3. Rectangle
4. Triangle
5. Trapezium
6. Pentagon
Enter your choice: 2
Enter the side: 2
Perimeter is: 8.0
-- program is finished running --
```



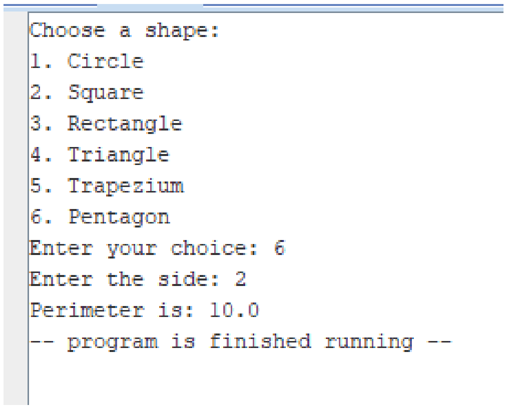
```
Choose a shape:
1. Circle
2. Square
3. Rectangle
4. Triangle
5. Trapezium
6. Pentagon
Enter your choice: 3
Enter the length: 2
Enter the width: 2
Perimeter is: 8.0
-- program is finished running --
```



```
Choose a shape:
1. Circle
2. Square
3. Rectangle
4. Triangle
5. Trapezium
6. Pentagon
Enter your choice: 4
Enter side a: 2
Enter side b: 2
Enter side c: 2
Perimeter is: 6.0
-- program is finished running --
```



```
Choose a shape:
1. Circle
2. Square
3. Rectangle
4. Triangle
5. Trapezium
6. Pentagon
Enter your choice: 5
Enter side a: 2
Enter side b: 2
Enter side c: 2
Enter side d: 2
Perimeter is: 8.0
-- program is finished running --
```



```
Choose a shape:
1. Circle
2. Square
3. Rectangle
4. Triangle
5. Trapezium
6. Pentagon
Enter your choice: 6
Enter the side: 2
Perimeter is: 10.0
-- program is finished running --
```