

Program 4

Snippet:

```
1  #calculate factorial of a number
2  .data
3  string1: .ascii "\nEnter a number of your choice:\n"
4  string2: .ascii "\nThe factorial is:\n"
5  string3: .ascii "\nThe number entered is less than 0\n"
6  .text
7  la $a0,string1
8  li $v0,4
9  syscall
10 #accept number from the user
11 li $v0,5
12 syscall
13 move $s0,$v0 #move the number into s0 register
14
15 #check condition
16 bltz $s0, error
17 move $a0, $s0 #pass the number as an argument to function
18
19 #call factorial
20 jal factorial
21 move $s1, $v0 #move value of $v0 into $s1
22
23 #print correct result
24 la $a0, string2
25 li $v0, 4 #to print strings
26 syscall
27
28 la $v0, 1
29 move $a0, $s1
30 syscall
31
32
33 # end program
34 li $v0, 10
35 syscall
36
37 error:
38 #print the wrong message if input is less than 0
39 la $a0, string3
40 li $v0, 4 #to print strings
41 syscall
42
43 # end program
44 li $v0, 10
45 syscall
46
47 factorial:
48 addi $sp, $sp, -8 #adjust stack for 2 items
49 sw $ra, 4($sp) #save return address
50 sw $a0, 0($sp) #save argument
51 slti $t0, $a0, 1 #test for n < 1
52 beq $t0, $zero, L1
53 addi $v0, $zero, 1 #if so, result is 1
54 addi $sp, $sp, 8 #pop 2 items from stack
55 jr $ra #and return
56
57 L1:
58 add $a0, $a0, -1 # else decrement n
59 jal factorial #recursive call
60 lw $a0, 0($sp) #restore original n
61 lw $ra, 4($sp) #and return address
62 addi $sp, $sp, 8 #pop 2 items from stack
63 mul $v0, $a0, $v0 #multiply to get result
64 jr $ra #and return
```

Output:

```
Enter a number of your choice:
**** user input : -3

The number entered is less than 0

-- program is finished running --
```

```
Enter a number of your choice:
**** user input : 5

The factorial is:
120

-- program is finished running --
```

Script:

#calculate factorial of a number

.data

string1: .ascii "\nEnter a number of your choice:\n"

string2: .ascii "\nThe factorial is:\n"

string3: .ascii "\nThe number entered is less than 0\n"

.text

la \$a0,string1

li \$v0,4

syscall

#accept number from the user

li \$v0,5

syscall

move \$s0,\$v0 #move the number into s0 register

#check condition

bltz \$s0, error

move \$a0, \$s0 #pass the number as an argument to function

#call factorial

jal factorial

move \$s1, \$v0 #move value of \$v0 into \$s1

#print correct result

la \$a0, string2

li \$v0, 4 #to print strings

syscall

la \$v0, 1

```
move $a0, $s1
```

```
syscall
```

```
# end program
```

```
li $v0, 10
```

```
syscall
```

```
error:
```

```
#print the wrong message if input is less than 0
```

```
la $a0, string3
```

```
li $v0, 4 #to print strings
```

```
syscall
```

```
# end program
```

```
li $v0, 10
```

```
syscall
```

```
factorial:
```

```
addi $sp, $sp, -8 #adjust stack for 2 items
```

```
sw $ra, 4($sp) #save return address
```

```
sw $a0, 0($sp) #save argument
```

```
slti $t0, $a0, 1 #test for n < 1
```

```
beq $t0, $zero, L1
```

```
addi $v0, $zero, 1 #if so, result is 1
```

```
addi $sp, $sp, 8 #pop 2 items from stack
```

```
jr $ra #and return
```

```
L1:
```

```
add $a0, $a0, -1 # else decrement n
jal factorial #recursive call
lw $a0, 0($sp) #restore original n
lw $ra, 4($sp) #and return address
addi $sp, $sp, 8 #pop 2 items from stack
mul $v0, $a0, $v0 #multiply to get result
jr $ra #and return
```