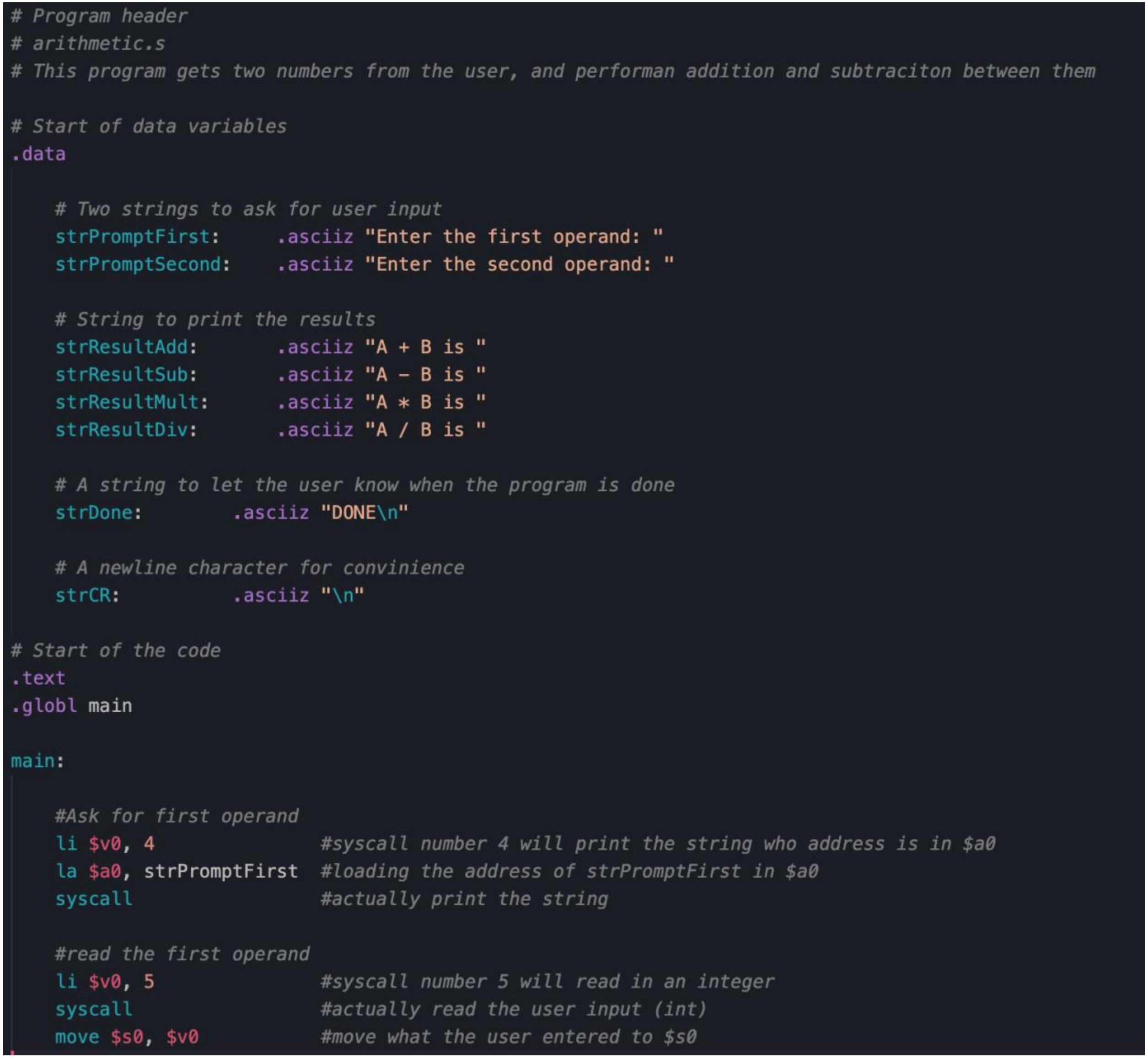
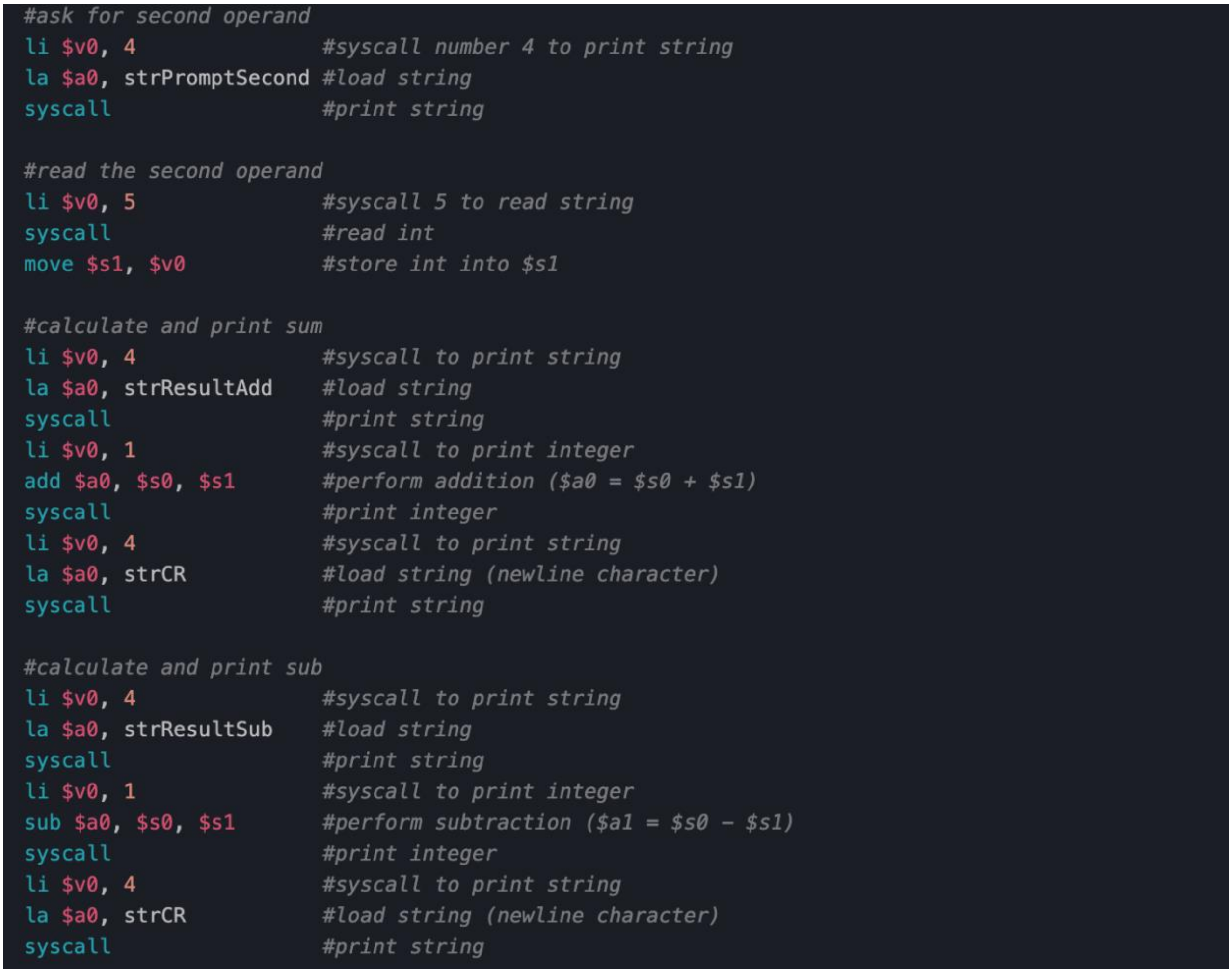
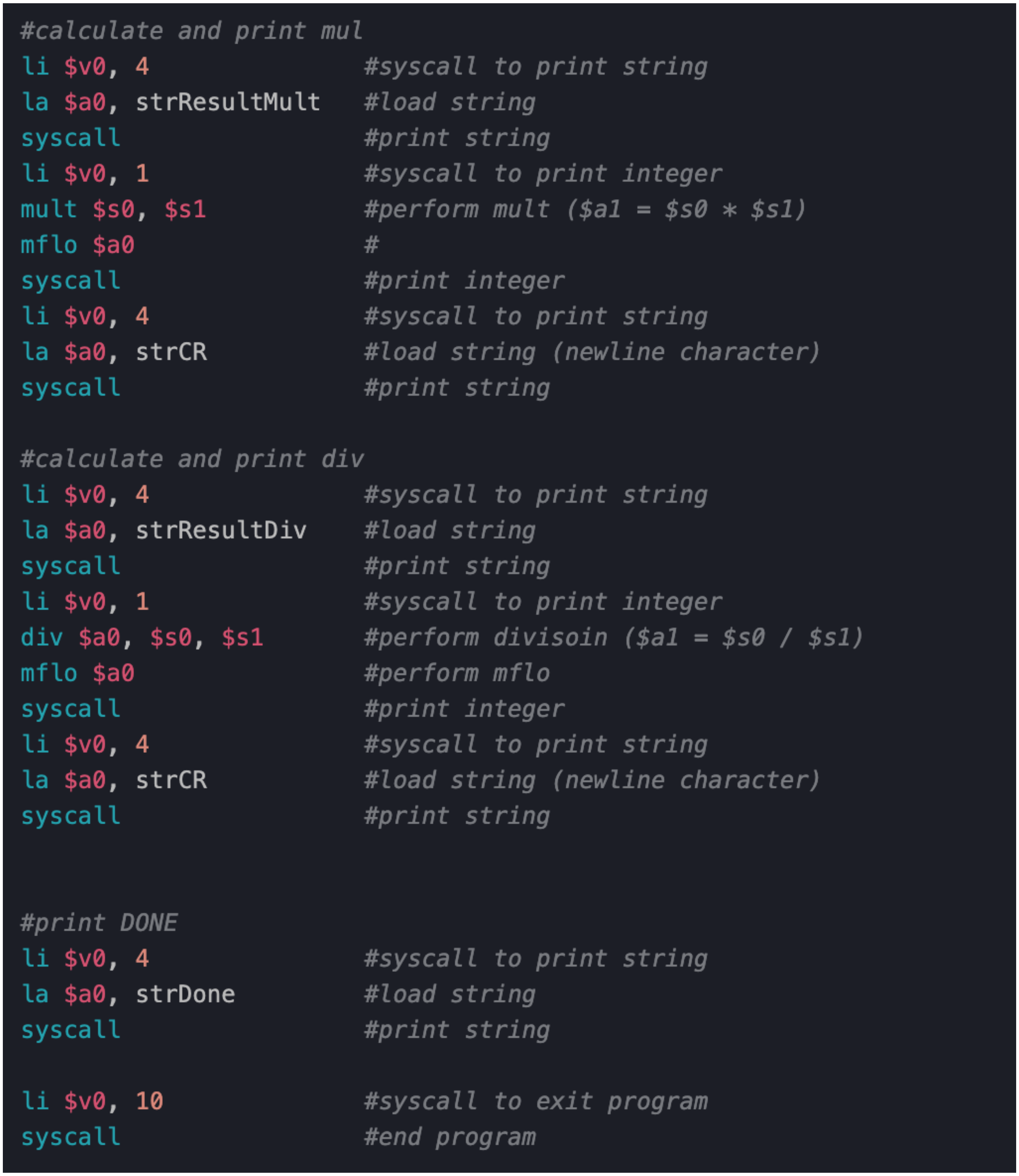
Assembly Code:



```
# Program header
# arithmetic.s
# This program gets two numbers from the user, and performan addition and subtraciton between them

# Start of data variables
.data

    # Two strings to ask for user input
    strPromptFirst:     .asciiz "Enter the first operand: "
    strPromptSecond:    .asciiz "Enter the second operand: "

    # String to print the results
    strResultAdd:       .asciiz "A + B is "
    strResultSub:       .asciiz "A - B is "
    strResultMult:      .asciiz "A * B is "
    strResultDiv:       .asciiz "A / B is "

    # A string to let the user know when the program is done
    strDone:        .asciiz "DONE\n"

    # A newline character for convinience
    strCR:          .asciiz "\n"

# Start of the code
.text
.globl main

main:

    #Ask for first operand
    li $v0, 4               #syscall number 4 will print the string who address is in $a0
    la $a0, strPromptFirst  #loading the address of strPromptFirst in $a0
    syscall                 #actually print the string

    #read the first operand
    li $v0, 5               #syscall number 5 will read in an integer
    syscall                 #actually read the user input (int)
    move $s0, $v0           #move what the user entered to $s0
```

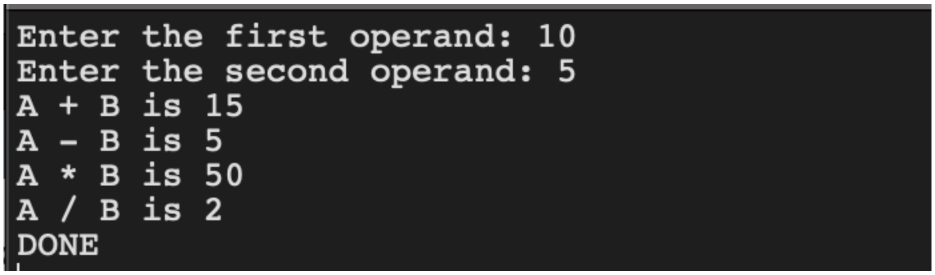


```
#ask for second operand
li $v0, 4               #syscall number 4 to print string
la $a0, strPromptSecond #load string
syscall                 #print string

#read the second operand
li $v0, 5               #syscall 5 to read string
syscall                 #read int
move $s1, $v0           #store int into $s1

#calculate and print sum
li $v0, 4               #syscall to print string
la $a0, strResultAdd    #load string
syscall                 #print string
li $v0, 1               #syscall to print integer
add $a0, $s0, $s1       #perform addition ($a0 = $s0 + $s1)
syscall                 #print integer
li $v0, 4               #syscall to print string
la $a0, strCR           #load string (newline character)
syscall                 #print string

#calculate and print sub
li $v0, 4               #syscall to print string
la $a0, strResultSub    #load string
syscall                 #print string
li $v0, 1               #syscall to print integer
sub $a0, $s0, $s1       #perform subtraction ($a1 = $s0 - $s1)
syscall                 #print integer
li $v0, 4               #syscall to print string
la $a0, strCR           #load string (newline character)
syscall                 #print string
```



```
#calculate and print mul
li $v0, 4               #syscall to print string
la $a0, strResultMult   #load string
syscall                 #print string
li $v0, 1               #syscall to print integer
mult $s0, $s1           #perform mult ($a1 = $s0 * $s1)
mflo $a0                #
syscall                 #print integer
li $v0, 4               #syscall to print string
la $a0, strCR           #load string (newline character)
syscall                 #print string

#calculate and print div
li $v0, 4               #syscall to print string
la $a0, strResultDiv    #load string
syscall                 #print string
li $v0, 1               #syscall to print integer
div $a0, $s0, $s1       #perform divisoin ($a1 = $s0 / $s1)
mflo $a0                #perform mflo
syscall                 #print integer
li $v0, 4               #syscall to print string
la $a0, strCR           #load string (newline character)
syscall                 #print string


#print DONE
li $v0, 4               #syscall to print string
la $a0, strDone         #load string
syscall                 #print string

li $v0, 10              #syscall to exit program
syscall                 #end program
```

Output:



```
Enter the first operand: 10
Enter the second operand: 5
A + B is 15
A - B is 5
A * B is 50
A / B is 2
DONE
```

List of Registers

- LO = 2 – Used to display the quotient for division and the output for multiplication
- R2 [v0] = a – Defines what is going to be read (int, string, etc)
- R4 [a0] = 1001005d – Used to load strings before printing them
- R16 [s0] = a – Used to store the first input
- R17 [s1] = 5 – Used to store the second input