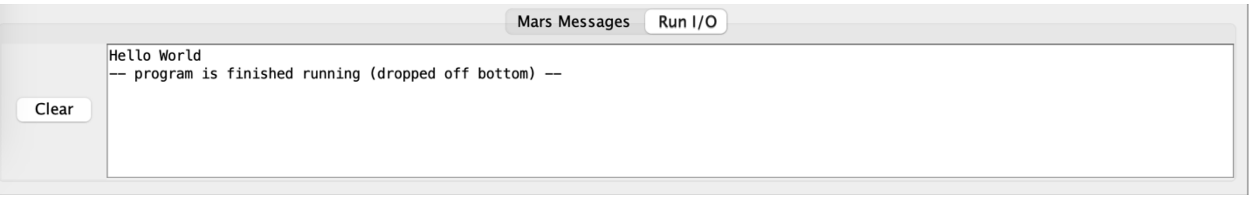
Week 5

Assignment 1

Source Code and explanation

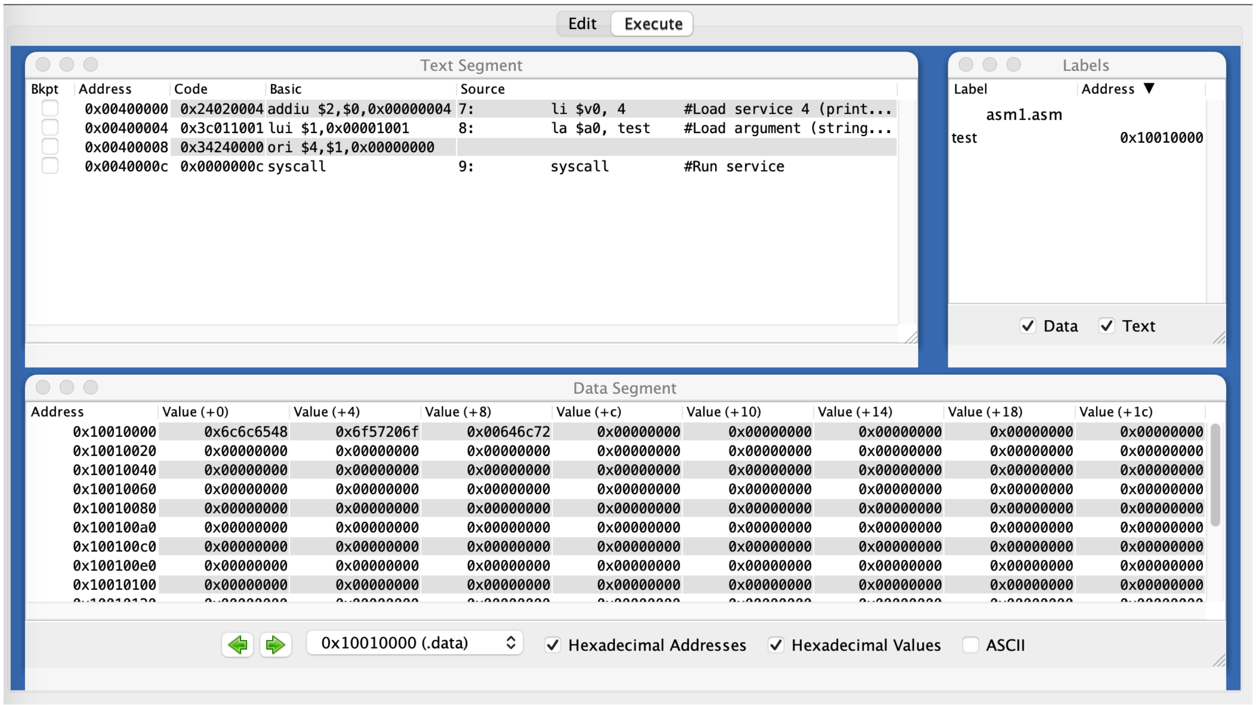
```
#Laboratory Exercise 5, Assignment 1

.data
      test: .asciiz "Hello World"

.text
      li $v0, 4           #Load service 4 (print string)
      la $a0, test        #Load argument (string at address)
      syscall             #Run service
```

Run results: Program outputs the string “Hello World” to standard output.



```
Hello World
-- program is finished running (dropped off bottom) --
```

Memory Allocation:



Bkpt	Address	Code	Basic	Source		
□	0x00400000	0x24020004	addiu $2,$0,0x00000004	7:	li $v0, 4	#Load service 4 (print...
□	0x00400004	0x3c011001	lui $1,0x00001001	8:	la $a0, test	#Load argument (string...
□	0x00400008	0x34240000	ori $4,$1,0x00000000			
□	0x0040000c	0x0000000c	syscall	9:	syscall	#Run service

Label	Address ▼
asm1.asm	
test	0x10010000

☑ Data ☑ Text

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x6c6c6548	0x6f57206f	0x00646c72	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

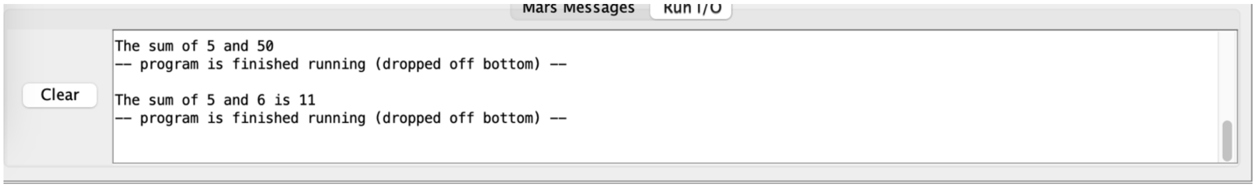
0x10010000 (.data) ☑ Hexadecimal Addresses ☑ Hexadecimal Values □ ASCII

Assignment 2

Source code and explanation

```
#Laboratory Exercise 5, Assignment 2

.data
      str0: .asciiz "The sum of "
      str1: .asciiz " and "
      str2: .asciiz " is "
.text
      li $s0, 5        #Init value for $s0 and $s1
      li $s1, 6


      li $v0, 4        #Load service 4 (print string)
      la $a0, str0           #Load str0 address to argument register
      syscall                #Call service to print "The sum of "

      li $v0, 1        #Load service 1 (print integer)
      addu $a0, $zero, $s0   #First operand s0 = 5
      syscall

      li $v0, 4        #Load service 4 (print string)
      la $a0, str1           #Load str1 address to argument register
      syscall                #Call service to print " and "

      li $v0, 1        #Load service 1 (print integer)
      addu $a0, $zero, $s1   #Second operand s1 = 6
      syscall

      li $v0, 4        #Load service 4 (print string)
      la $a0, str2           #Load str1 address to argument register
      syscall                #Call service to print " is "

      li $v0, 1        #Load service 1 (print integer)
      add $a0, $s0, $s1#Second operand s1 = 6
      syscall
```

Run results: Output integer 11 as sum of 5 and 6



```
The sum of 5 and 50
-- program is finished running (dropped off bottom) --

The sum of 5 and 6 is 11
-- program is finished running (dropped off bottom) --
```

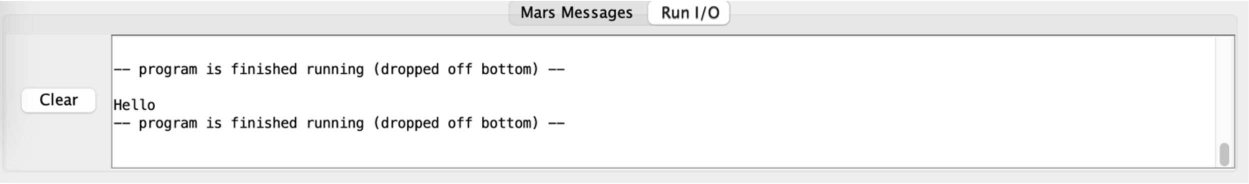
Assignment 3

Source code and explanation

```
#Laboratory Exercise 5, Assignment 3
.data
      x: .space 100          #Allocate empty space for string X
      y: .asciiz "Hello"     #Init string Y
```

```
.text

strcpy:
      li $s0, 0        #Init i = 0 (address offset value)
      la $a1, y        #Load address of y[0] to $a1
      la $a0, x        #Load address of x[0] to $a0
      Loop:
            add $t1,$s0,$a1 #Advance address of y (t1 = y[i+1])
            lb $t2,0($t1)    #Load byte value of y to $t2

            add $t3,$s0,$a0 #Advance address of x (t3 = x[i+1])
            sb $t2,0($t3)    #Store byte value of x from $t2

            beq $t2,$zero,endstrcpy      #If y[i]==0, exit loop
            addi $s0,$s0,1         #Increase address offset
            j Loop
            nop


endstrcpy:
      li $v0, 4        #Print result
      la $a0, x
      syscall
```

Run result:



```
-- program is finished running (dropped off bottom) --

Hello
-- program is finished running (dropped off bottom) --
```

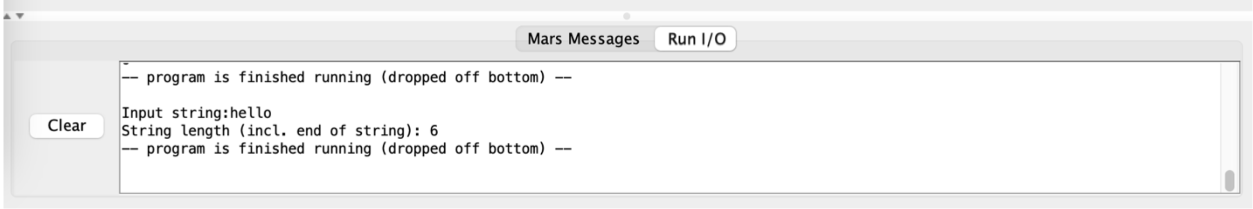
Assignment 4

Source code and explanation

```
#Laboratory Exercise 5, Home Assignment 4

.data
      str: .space 50
      msg1: .asciiz "Input string:,Äù
      msg2: .asciiz "String length (incl. end of string): "

.text
main:
get_string:
      li $v0, 4        #Load service 4 print string
      la $a0, msg1
      syscall

      li $v0, 8        #Load service 8 read string
      la $a0, str      #Load address of str
      li $a1, 50       #Maximum characters to read 50
      syscall
```

```
get_length:

      add $t3, $zero, $zero  #Current string length
      add $t0, $zero, $zero  #i = 0

      check_char:
            add $t1, $a0, $t0      #Advance address pointer
            lb $t2, 0($t1)         #Load current byte value to check for
NULL
            beq $t2,$zero,end_of_str#Check for NULL char to end loop
            addi $t3, $t3, 1 #length = length + 1
            addi $t0, $t0, 1 #i = i + 1 (increase address pointer)
            j check_char
end_of_str:
end_of_get_length:
print_length:
      li $v0, 4         #Load service 4 print string
      la $a0, msg2
      syscall

      li $v0, 1         #Load service 1 print integer
      add $a0, $zero, $t3    #Load length
      syscall
```

Run results:



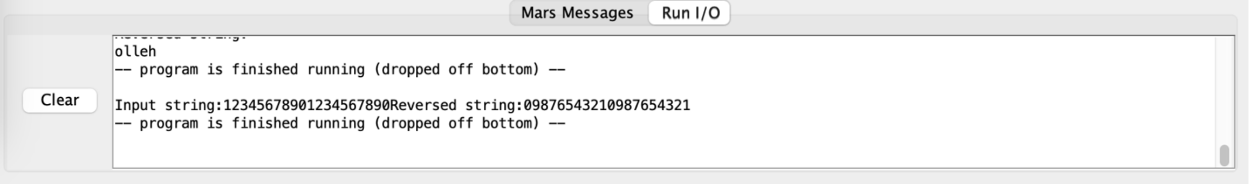
```
-- program is finished running (dropped off bottom) --

Input string:hello
String length (incl. end of string): 6
-- program is finished running (dropped off bottom) --
```

Assignment 5

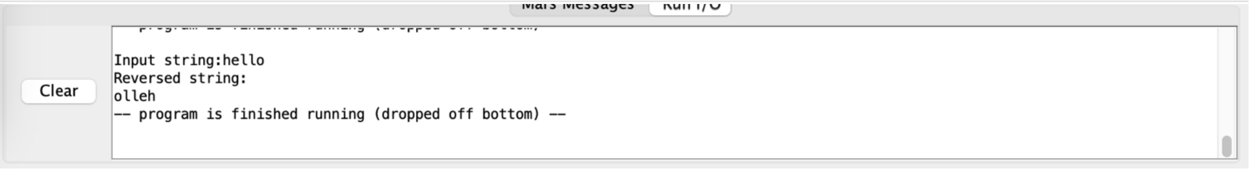
Source code and explanation:

```
#Laboratory Exercise 5, Home Assignment 5

.data
      msg: .asciiz "Input string:"
      str: .space  21        #Allocate space for string
      str_rev: .space  21    #Allocate space for reversed string
      reverse_msg: .asciiz "Reversed string:"

.text
main:
      li $v0, 4   #Promt input
      la $a0, msg
      syscall

      li $v0, 8   #Read string
      la $a0, str
      li $a1, 21       #Maximum input length 20 + NULL
      syscall

get_length:      # Find length of string
      li $t0, 0       #Init i = 0 for loop
get_length_loop:
      lb $t1, str($t0)       #Load current byte value to check for NULL
      beqz $t1, end_get_length     #Check for NULL char to end loop
      addi $t0, $t0, 1       #Increment address counter i = i + 1
      b get_length_loop
end_get_length:

get_reverse:
      addi $t0, $t0, -1      #Decrement address counter i = i - 1 to skip
NULL
      li $t2, 0                    #Index for str_rev
reverse_loop:

      lb $t1, str($t0)       #Load byte from original string
      sb $t1, str_rev($t2)         #Store byte in reversed string
      beq $t0, $zero, print_reverse      #If i = 0, exit loop
      addi $t0, $t0, -1            #Decrement i = i - 1
      addi $t2, $t2, 1       #Increment index for str_rev
      b reverse_loop
print_reverse:

    li $v0, 4
    la $a0, reverse_msg
    syscall

    la $a0, str_rev    #Print reversed string
    li $v0, 4
    syscall
```

Run results:



```
Reversed string:
olleh
-- program is finished running (dropped off bottom) --

Input string:12345678901234567890Reversed string:09876543210987654321
-- program is finished running (dropped off bottom) --
```



```
-- program is finished running (dropped off bottom) --

Input string:hello
Reversed string:
olleh
-- program is finished running (dropped off bottom) --
```