

Laboratory Exercise 5

Character string with SYSCALL function, and sorting

- Assignment 1:** Create a new project to implement the program in Home Assignment 1. Compile and upload to simulator. Run and observe the result. Go to data memory section, check how test string are stored and packed in memory.

The screenshot displays the Mars MIPS simulator interface. The main window is divided into several sections:

- Text Segment:** Shows assembly code with columns for Bkpt, Address, Code, Basic, and Source. The code includes instructions like `addiu $2,$0,0x00000004`, `li $v0, 4`, `lui $1,0x00001001`, `la $a0, test`, `ori $4,$1,0x00000000`, and `syscall`.
- Data Segment:** A table showing memory addresses and their corresponding values in hexadecimal. The address `0x10010000` contains the value `0x6c6c6548`, which corresponds to the string "Hello World" in ASCII.
- Registers:** A table listing MIPS registers (e.g., \$zero, \$at, \$v0, \$v1, \$a0, \$a1, etc.) and their current values. The register `$a0` is highlighted with a value of `0x10010000`.
- Mars Messages:** A section at the bottom showing the output of the program, which is "Hello World".

Test có địa chỉ `0x10010000` có value là "Hello World\0"

- Chương trình load 4 vào `$v0` để chạy lệnh `print`. Hai cặp lệnh `lui` và `ori` để gán `$a0` = địa chỉ của test.
- `Syscall` in giá trị của `$a0`

2. Assignment 2: Create a new project to print the sum of two register \$s0 and \$s1 according to this format: “The sum of (s0) and (s1) is (result)”

The screenshot displays the Mars MIPS simulator interface. The main window shows the assembly code for a file named `mips2.asm`. The code is as follows:

```

1  .data          ## Data declaration section
2  a1: .asciiz "The sum of "
3  a2: .asciiz " and "
4  a3: .asciiz " is "
5  .text          ## Assembly language instructions go in text segment
6  main:          ## Start of code section
7      addi $s0, $zero, 1      # s0 = 1
8      addi $s1, $zero, 2      # s1 = 2
9      add $s2, $s0, $s1       # s2 = s0 + s1
10     li $v0, 4               # system call code to print string from a1
11     la $a0, a1
12     syscall                # call operating system to perform operation
13     li $v0, 1               # system call code to print integer from $s0
14     move $a0, $s0
15     syscall                # call operating system to perform operation
16     li $v0, 4               # system call code to print string from a2
17     la $a0, a2
18     syscall
19     li $v0, 1               # system call code to print integer from $s1
20     move $a0, $s1
21     syscall                # call operating system to perform operation
22     li $v0, 4               # system call code to print string from a3
23     la $a0, a3
24     syscall
25     li $v0, 1               # system call code to print integer from $s2
26     move $a0, $s2
27     syscall
  
```

Below the code, the status bar indicates "Line: 8 Column: 32" and "Show Line Numbers" is checked. The "Mars Messages" window shows the output of the program:

```

The sum of 1 and 2 is 3
-- program is finished running (dropped off bottom) --
  
```

The "Registers" window on the right shows the state of the MIPS registers. The registers are organized into three columns: Name, Number, and Value. The registers are as follows:

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x00000001
\$v1	3	0x00000000
\$a0	4	0x00000003
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000001
\$s1	17	0x00000002
\$s2	18	0x00000003
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7fffffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400060
hi		0x00000000
lo		0x00000000

Với trường hợp 1 khi cho $s0 = 1$ và $s1 = 2$, cần in ra $s2 = s0 + s1$ theo dự kiến = 3 thì chương trình đã chạy đúng, Run I/O hiện ra “The sum of 1 and 2 is 3”. Thử với trường hợp khác khi $s0 = 17$ và $s1 = 15$, theo dự kiến = 32 thì chương trình tiếp tục chạy đúng ra kết quả “The sum of 17 and 15 is 32” như sau đây.

EditExecute

mips2.asm

```

1 .data          ## Data declaration section
2 a1: .asciiz "The sum of "
3 a2: .asciiz " and "
4 a3: .asciiz " is "
5 .text          ## Assembly language instructions go in text segment
6 main:          ## Start of code section
7     addi $s0,$zero,17      # s0 = 17
8     addi $s1,$zero,15      # s1 = 15
9     add $s2,$s0,$s1        # s2 = s0 + s1
10    li $v0, 4              # system call code to print string from a1
11    la $a0, a1
12    syscall              # call operating system to perform operation
13    li $v0, 1              # system call code to print integer from $s0
14    move $a0, $s0
15    syscall              # call operating system to perform operation
16    li $v0, 4              # system call code to print string from a2
17    la $a0, a2
18    syscall
19    li $v0, 1              # system call code to print integer from $s1
20    move $a0, $s1
21    syscall              # call operating system to perform operation
22    li $v0, 4              # system call code to print string from a3
23    la $a0, a3
24    syscall
25    li $v0, 1              # system call code to print integer from $s2
26    move $a0, $s2
27    syscall

```

Line: 8 Column: 34 ☒ Show Line Numbers

Mars MessagesRun I/O

The sum of 1 and 2 is 3

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

Clear

The sum of 17 and 15 is 32

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

RegistersCoprocc 1Coprocc 0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x00000001
\$v1	3	0x00000000
\$a0	4	0x00000020
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000011
\$s1	17	0x0000000f
\$s2	18	0x00000020
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffcfc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400060
hi		0x00000000
lo		0x00000000

- Assignment 3:** Create a new project to implement the program in Home Assignment 2. Add more instructions to assign a test string for y variable and implement strcpy function. Compile and upload to simulator. Run and observe the result.

EditExecute

mips1.asm

mips2.asm

mips3.asm

```

1  #Laboratory Exercise 5, Home Assignment 2
2  .data
3  x: .space 32          # destination string x, empty
4  y: .asciiz "Hello"    # source string y
5  .text
6  init:
7      la $a1,y          #load address of y[0] to $a1
8      la $a0,x          #load address of x[0] to $a0
9  strcpy:
10     add $s0,$zero,$zero # $s0 = i = 0
11  L1:
12     add $t1,$s0,$a1     # $t1 = $s0 + $a1 = i + y[0]
13                     # = address of y[i]
14     lb $t2,0($t1)       # $t2 = value at $t1 = y[i]
15     add $t3,$s0,$a0     # $t3 = $s0 + $a0 = i + x[0]
16                     # = address of x[i]
17     sb $t2,0($t3)       # x[i] = $t2 = y[i]
18     beq $t2,$zero,end_of_strcpy # if y[i] == 0, exit
19     nop
20     addi $s0,$s0,1      # $s0 = $s0 + 1 <-> i = i + 1
21     j L1               # next character
22     nop
23 end_of_strcpy:
24     li $v0, 4          # system call code to print string from ax
25     la $a0, x
26     syscall            # call operating system to perform operation
27

```

Line: 1 Column: 1 ☒ Show Line Numbers

Mars MessagesRun I/O

Hello

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

Clear

Registers

Coproc 1

Coproc 0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x00000004
\$v1	3	0x00000000
\$a0	4	0x10010000
\$a1	5	0x10010020
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x10010025
\$t2	10	0x00000000
\$t3	11	0x10010005
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000005
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$s8	24	0x00000000
\$s9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400048
hi		0x00000000
lo		0x00000000

Khi cho chuỗi y là “Hello” với lệnh strcpy thì chương trình hiện đúng ra là “Hello”
 Thử với chuỗi dài hơn: “Hello! This is Assignment 3 of Laboratory Week 5.”

EditExecute

mips1.asm

mips2.asm

mips3.asm

```

1  #Laboratory Exercise 5, Home Assignment 2
2  .data
3  x: .space 32          # destination string x, empty
4  y: .asciiz "Hello! This is Assignment 3 of Laboratory Week 5." # source string y
5  .text
6  init:
7      la $a1,y          #load address of y[0] to $a1
8      la $a0,x          #load address of x[0] to $a0
9  strcpy:
10     add $s0,$zero,$zero # $s0 = i = 0
11  L1:
12     add $t1,$s0,$a1     # $t1 = $s0 + $a1 = i + y[0]
13                     # = address of y[i]
14     lb $t2,0($t1)       # $t2 = value at $t1 = y[i]
15     add $t3,$s0,$a0     # $t3 = $s0 + $a0 = i + x[0]
16                     # = address of x[i]
17     sb $t2,0($t3)       # x[i] = $t2 = y[i]
18     beq $t2,$zero,end_of_strcpy # if y[i] == 0, exit
19     nop
20     addi $s0,$s0,1      # $s0 = $s0 + 1 <-> i = i + 1
21     j L1               # next character
22     nop
23 end_of_strcpy:
24     li $v0, 4           # system call code to print string from ax
25     la $a0, x
26     syscall            # call operating system to perform operation
27

```

Line: 4 Column: 68 ☒ Show Line Numbers

Mars MessagesRun I/O

Hello! This is Assignment 3 of Laboratory Week 5.

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

Clear

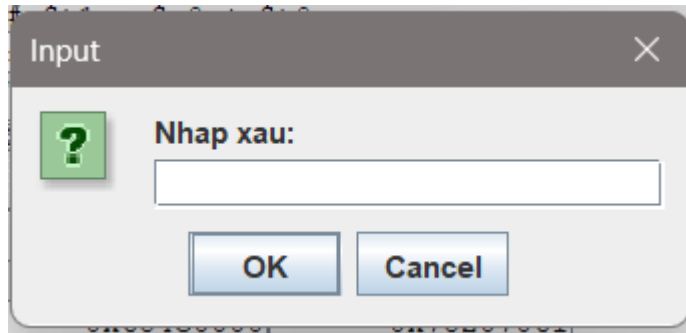
RegistersCoprocc0Coprocc1

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x00000004
\$v1	3	0x00000000
\$a0	4	0x10010000
\$a1	5	0x10010020
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x10010051
\$t2	10	0x00000000
\$t3	11	0x10010031
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000031
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffeffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400048
hi		0x00000000
lo		0x00000000

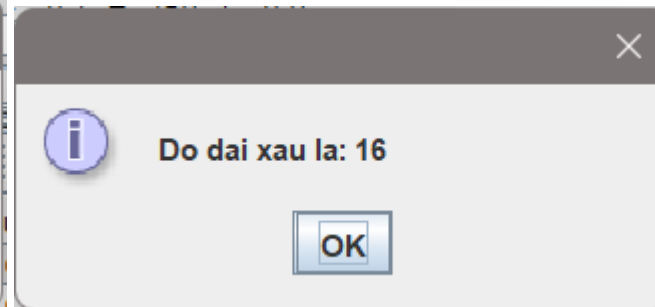
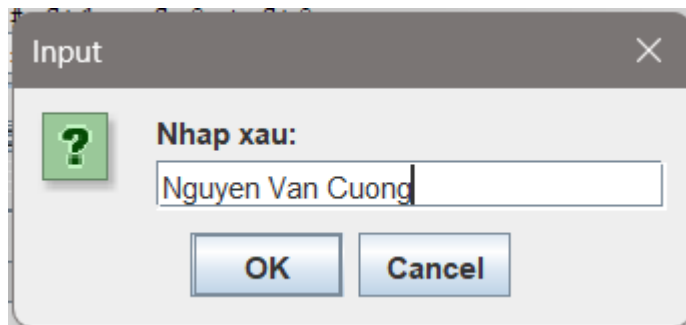
Run I/O vẫn hiện đúng ra chuỗi “Hello! This is Assignment 3 of Laboratory Week 5.”

- Assignment 4:** Accomplish the Home Assignment 3 with syscall function to get a string from dialog and show the length to message dialog.

Edit	Execute
<div>mips1.asm mips2.asm mips3.asm mips4.asm</div> <pre>1 #Laboratory Exercise 5, Home Assignment 3 2 .data 3 string: .space 50 4 Message1: .ascii "Nhap xau: " 5 Message2: .ascii "Do dai xau la: " 6 .text 7 main: 8 get_string: # TODO 9 li \$v0, 54 10 la \$a0, Message1 11 la \$a1, string 12 la \$a2, 50 13 syscall 14 get_length: 15 la \$a0, string # \$a0 = address(string[0]) 16 add \$t0, \$zero, \$zero # \$t0 = i = 0 17 check_char: 18 add \$t1, \$a0, \$t0 # \$t1 = \$a0 + \$t0 19 # = address(string[i]) 20 lb \$t2, 0(\$t1) # \$t2 = string[i] 21 beq \$t2, \$zero, end_of_str # is null char? 22 addi \$t0, \$t0, 1 # \$t0 = \$t0 + 1 -> i = i + 1 23 j check_char 24 end_of_str: 25 end_of_get_length: 26 addi \$t0, \$t0, -1 27 print_length: 28 li \$v0, 56 29 la \$a0, Message2 30 move \$a1, \$t0 31 syscall</pre>	



Thử nhập xâu “Nguyen Van Cuong”, dự kiến kết quả độ dài xâu sẽ là: 16



Chương trình đúng

5. **Assignment 5:** Write a program that let user input a string by typing individual letters. Input process will be terminated when user press Enter or then length of the string exceed 20 characters. Print the reverse string.

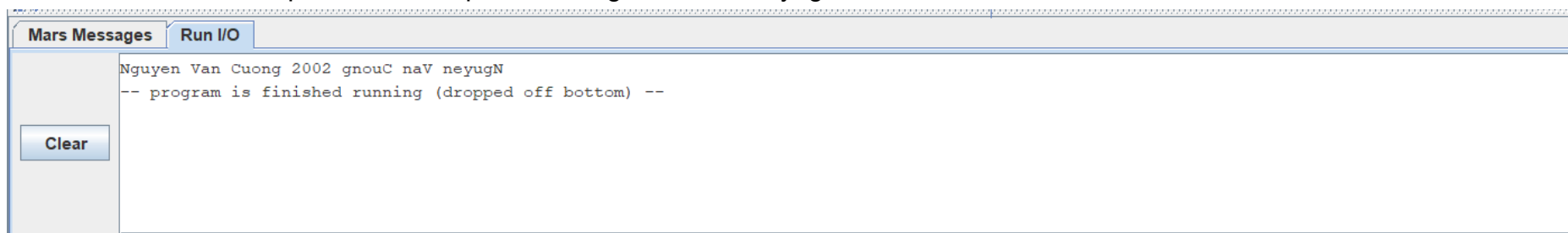
```

1  .data
2  string: .space 20
3  reverse: .space 20
4  .text
5  get_string:
6      li $v0, 8
7      la $a0, string
8      li $a1, 20
9      syscall
10 get_length:
11     la $a0, string # a0 = Address(string[0])
12     li $v0, 0 # v0 = length = 0
13
14 check_char:
15     add $t1,$a0,$t0      # $t1 = $a0 + $t0
16                          # = address(string[i])
17     lb $t2, 0($t1)      # $t2 = string[i]
18     beq $t2, $zero, end_of_str # is null char?
19     addi $t0, $t0, 1     # $t0 = $t0 + 1 -> i = i + 1
20     j check_char
21 end_of_str:
22 end_of_get_length:
23     add $t1,$a0,$t0
24     li $t2, 0           # counter i = 0
25     li $v0, 11
26 reverseString:
27     slt $t3, $t2, $t0   # if i < stringlength
28     beq $t3, $0, Exit   # if t3 reaches he end of the array
29     addi $t1, $t1, -1   # decrement the array
30     addi $t2,$t2,1
31     lbu $a0, 0($t1)     # load the array from the end
32     syscall
33     j reverseString
34 Exit:

```

Chương trình yêu cầu nhập xâu #get_string, thử nhập: “van cuong”, kết quả dự kiến phải ra: “gnouc nav”

Thử với xâu dài hơn: “Nguyen Van Cuong 20215006”. Vì xâu này có độ dài 26 kí tự bao gồm kí tự “\0” mà độ dài xâu yêu cầu là 20 kí tự nên kết quả dự kiến sẽ phải là: “02 gnouC naV neyugN”

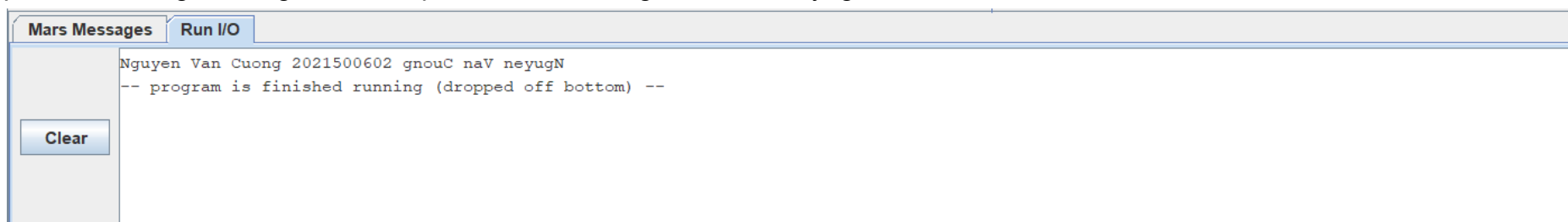


The screenshot shows a window titled "Mars Messages" with a "Run I/O" button. The output text is:

```
Nguyen Van Cuong 2002 gnouC naV neyugN
-- program is finished running (dropped off bottom) --
```

There is a "Clear" button on the left side of the window.

Tuy nhiên, thực tế do độ dài kí tự yêu cầu nhập chỉ là 20 nên lúc em nhập xâu, khi chương trình nhận đủ 20 kí tự sẽ tự động hiện ra kết quả. Có 1 cách tricky để nhập được xâu “Nguyen Van Cuong 20215006” là sẽ copy trước xâu này ở ngoài và paste vào trong chương trình, kết quả vẫn sẽ ra: “02 gnouC naV neyugN”



The screenshot shows a window titled "Mars Messages" with a "Run I/O" button. The output text is:

```
Nguyen Van Cuong 2021500602 gnouC naV neyugN
-- program is finished running (dropped off bottom) --
```

There is a "Clear" button on the left side of the window.