

Thực hành kiến trúc máy tính

Tuần 5

Bùi Quang Hưng – 20225849

Assignment 1

Bài làm

#Laboratory Exercise 5, Assignment 1

.data

test: .asciiz "Bui Quang Hung bi om roi huhu"

.text

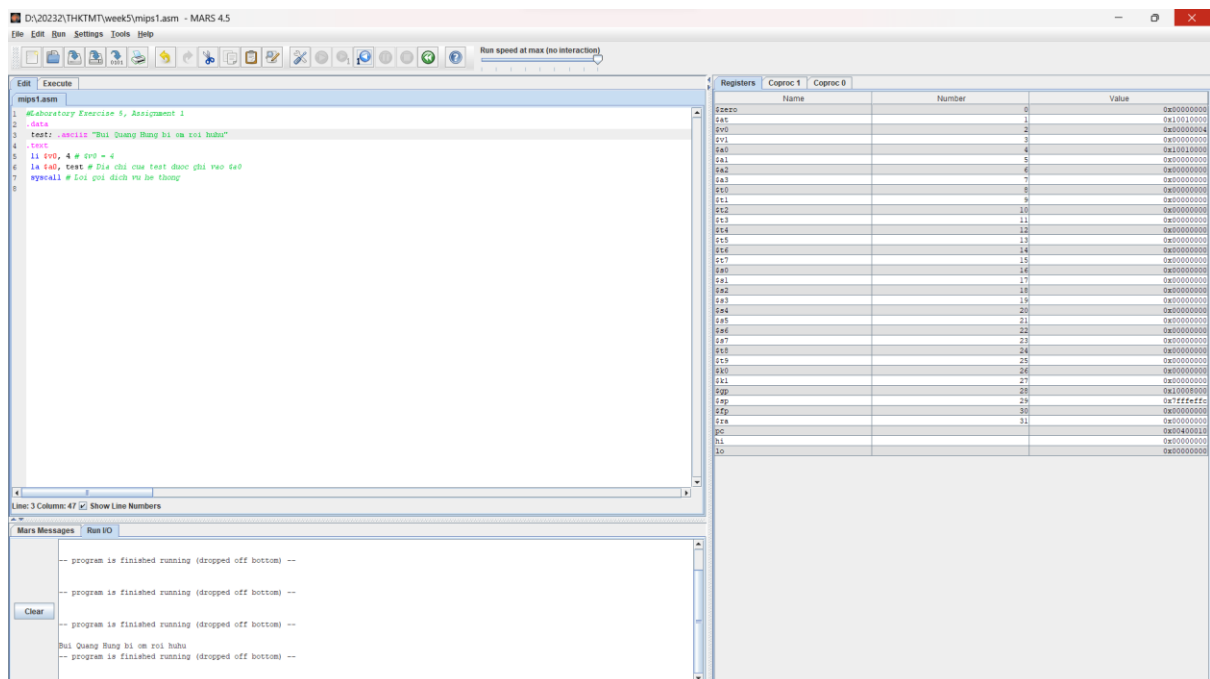
li \$v0, 4 # \$v0 = 4

la \$a0, test # Địa chỉ của test được ghi vào \$a0

syscall # Gọi gọi dịch vụ hệ thống

- Thanh ghi \$v0 được nạp giá trị 4 → system sẽ được thực hiện lệnh print string → in ra màn hình giá trị của test.

- \$a0 được nạp địa chỉ của chuỗi kí tự được lưu ở biến test.



→ Kết quả đúng với lý thuyết.

Assignment 2

Bài làm

#Laboratory Exercise 5, Assignment 2

.data

str1: .asciiz "The sum of "

str2: .asciiz " and "

str3: .asciiz " is "

.text

li \$s0, 1 # number1 = 1

li \$s1, 2 # number2 = 2

add \$t0, \$s0, \$s1 # \$t0 = Sum of 1 and 2

Print string "str1"

li \$v0, 4

la \$a0, str1

syscall

Print \$s0

li \$v0, 1

la \$a0, 0(\$s0)

syscall

Print string "str2"

li \$v0, 4

la \$a0, str2

syscall

Print \$s1

li \$v0, 1

la \$a0, 0(\$s1)

syscall

```
# Print string "str3"
```

```
li $v0, 4
```

```
la $a0, str3
```

```
syscall
```

```
# Print $t0
```

```
li $v0, 1
```

```
la $a0, 0($t0)
```

```
syscall
```

Exit: li \$v0, 10

Syscall

- Sử dụng lệnh syscall với lệnh 4 và 1 để in ra string và integer.
- Khởi tạo hai giá trị số nguyên và tổng của chúng

```
li $s0, 1 # number1 = 1
li $s1, 2 # number2 = 2
add $t0, $s0, $s1 # $t0 = Sum of 1 and 2
```

- In ra xâu “The sum of”

```
# Print string "str1"
li $v0, 4
la $a0, str1
syscall
```

- In ra số nguyên thứ nhất có giá trị là 1

```
# Print $s0
li $v0, 1
la $a0, 0($s0)
syscall
```

- In ra xâu “and”

```
# Print string "str2"
li $v0, 4
la $a0, str2
syscall
```

- In ra số nguyên thứ hai với giá trị là 2

```
# Print $s1
li $v0, 1
la $a0, 0($s1)
syscall
```

- In ra xâu “is”

```
# Print string "str3"
li $v0, 4
la $a0, str3
syscall
```

- Kết quả ra màn hình, cụ thể là $1 + 2 = 3$

```
# Print $t0
li $v0, 1
la $a0, 0($t0)
syscall
```

- Result:

```
The sum of 1 and 2 is 3
-- program is finished running --
```

➔ Kết quả đúng với lí thuyết

Assignment 3

Bài làm

#Laboratory Exercise 5, Assignment 3

.data

x: .space 32 # destination string x, empty

y: .asciiz "Bui Quang Hung ahihi" # source string y

.text

strcpy:

add \$s0,\$zero,\$zero # \$s0 = i = 0

la \$a1, y # Load address of y to \$a1

la \$a0, x # Load address of x to \$a0

L1:

add \$t1,\$s0,\$a1 # \$t1 = \$s0 + \$a1 = i + y[0]

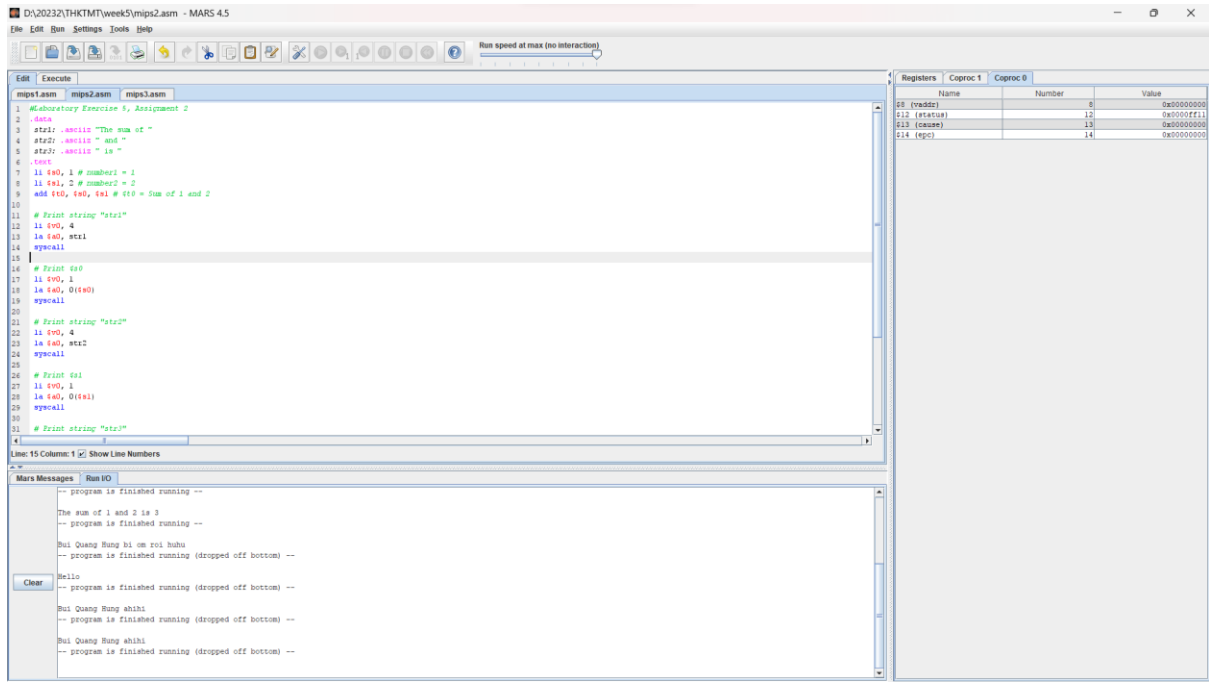
= address of y[i]

```

lb $t2,0($t1) # $t2 = value at $t1 = y[i]
add $t3,$s0,$a0 # $t3 = $s0 + $a0 = i + x[0]
                # = address of x[i]
sb $t2,0($t3) # x[i]= $t2 = y[i]
beq $t2,$zero,end_of_strcpy # if y[i] == 0, exit
nop
addi $s0,$s0,1      # $s0 = $s0 + 1 <-> i = i + 1
j L1    # next character
nop
end_of_strcpy:
printstring:
    li $v0, 4
    syscall

```

- Ý tưởng bài toán: Theo mỗi khối 4 bytes, các kí tự được copy từ thanh ghi \$a1 sang thanh ghi \$a0 theo chiều từ phải qua trái. Khi kết thúc thì sẽ chạy end_of_string, còn không thì tiếp tục với label L1 bằng cách sử dụng lệnh jump và lệnh beq
- Đầu tiên lưu giá trị của thanh ghi \$t1 là địa chỉ của y[i]
- Sau đó lưu giá trị của thanh ghi \$t1 vào thanh ghi \$t2
- Lưu giá trị của thanh ghi \$t3 là địa chỉ của x[i]
- Store giá trị của thanh ghi \$t2 vào địa chỉ đã lưu ở thanh ghi \$t3 (x[i] = t2 = y[i])
- Sử dụng lệnh beq để check xem thanh ghi \$t2 có rỗng hay không, nếu rỗng tức là string đã hết => exit
- Nếu không rỗng thì tăng index thêm 1 và lặp lại quá trình trên.
- Result:



→ Kết quả đúng với lý thuyết

Assignment 4

Bài làm

#Laboratory Exercise 5, Assignment 4

.data

string: .space 50

Message1: .asciiz "Nhap xau: "

Message2: .asciiz "Do dai xau la: "

.text

main:

get_string:

li \$v0, 54 # Get a string from dialog

la \$a0, Message1 # Load address of the Message1 to \$a0

la \$a1, string # Load address of input buffer "string" to \$a1

la \$a2, 50 # Maximum number of characters to read = 50

syscall

get_length:

la \$a0, string # \$a0 = address(string[0])

```
add $t0,$zero,$zero # $t0 = i = 0
```

check_char:

```
add $t1,$a0,$t0 # $t1 = $a0 + $t0
```

```
# = address(string[i])
```

```
lb $t2, 0($t1) # $t2 = string[i]
```

```
beq $t2, $zero, end_of_str # is null char?
```

```
addi $t0, $t0, 1 # $t0 = $t0 + 1 -> i = i + 1
```

```
j check_char
```

end_of_str:

end_of_get_length:

print_length:

```
addi $t0, $t0, -1
```

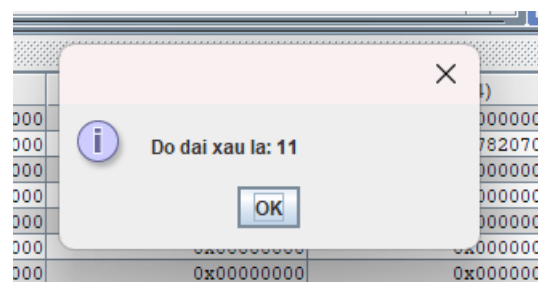
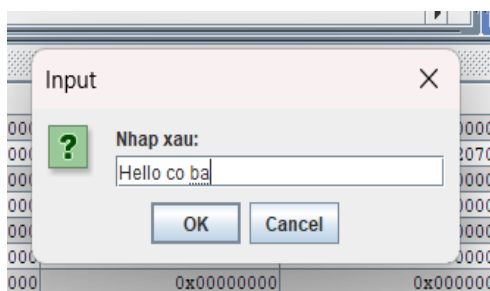
```
li $v0, 56
```

```
la $a0, Message2
```

```
la $a1, 0($t0)
```

```
syscall
```

- Result:



➔ Kết quả đúng với lí thuyết

Assignment 5

Bài làm

#Laboratory Exercise 5, Assignment 5

.data

```
string: .asciiz "Nhập kí tự "
```

```
string2:.asciiz "Chuoì dao nguoc la "
```

```
max: .space 20
```

```
str: .asciiz "\n"
```

```
.text
```

```
li $s0, 20      # N = 20
```

```
li $s1, 0       # i = 0
```

```
la $s2, max     # Load address of get_char[0]
```

```
li $s3, 10      # Char \n in ASCII
```

```
read_char:
```

```
beq $s1, $s0, end_read_char # i = N branch to exit
```

```
li $v0, 4
```

```
la $a0, string
```

```
syscall
```

```
li $v0, 12      #Doc vao tung ki tu
```

```
syscall
```

```
move $t0, $v0
```

```
beq $t0, $s3, end_read_char #"Neu ki tu nhap vao la dau enter thi dung"
```

```
li $v0, 4
```

```
la $a0, str
```

```
syscall
```

```
add $s5, $s2, $s1 # $s5 = Address of get_char[i] = get_char[0] + i
```

```
sb $t0, 0($s5)   #Store character to get_char[i]
```

```
addi $s1, $s1, 1 # i++
```

```
j read_char
```

```
end_read_char:
```

```
li $v0, 4
```

```
la $a0, string2
```

```
syscall
```

```
print:
```



```

li $v0, 11

lb $a0, 0($s5)

syscall

beq $s5, $s2, exit

addi $s5, $s5, -1      #in nguoc xau

j print

```

exit:

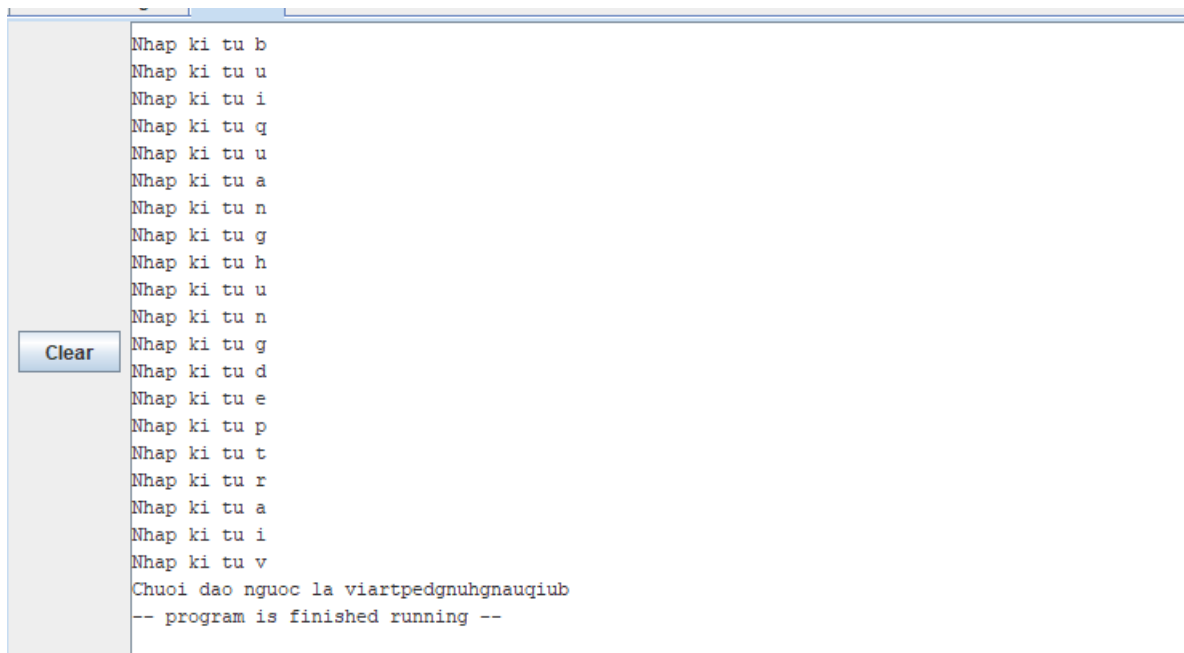
```

li $v0, 10

syscall

```

- Trường hợp nhập đủ 20 kí tự:

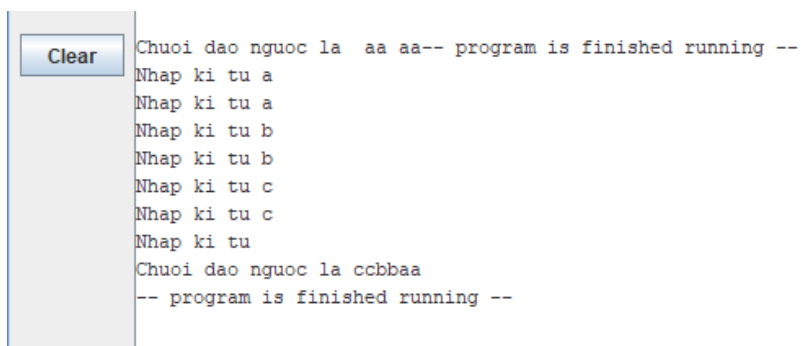


```

Nhap ki tu b
Nhap ki tu u
Nhap ki tu i
Nhap ki tu q
Nhap ki tu u
Nhap ki tu a
Nhap ki tu n
Nhap ki tu g
Nhap ki tu h
Nhap ki tu u
Nhap ki tu n
Nhap ki tu g
Nhap ki tu d
Nhap ki tu e
Nhap ki tu p
Nhap ki tu t
Nhap ki tu r
Nhap ki tu a
Nhap ki tu i
Nhap ki tu v
Chuoi dao nguoc la viartpedgnuhgnauqiub
-- program is finished running --

```

- Trường hợp chưa đến 20 kí tự:



```

Nhap ki tu a
Nhap ki tu a
Nhap ki tu b
Nhap ki tu b
Nhap ki tu c
Nhap ki tu c
Nhap ki tu
Chuoi dao nguoc la ccbbaa
-- program is finished running --

```

→ Kết quả đúng với lí thuyết

