

BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH – TUẦN 5

Họ và tên: Nguyễn Mạnh Tùng

MSSV: 20225682

Assignment 1:

- Code:

```
.data
```

```
test: .asciiz "Nguyen_Manh_Tung_VN07_K67"
```

```
.text
```

```
li $v0, 4          # lựa chọn chức năng
```

```
la $a0, test        # cập nhật dữ liệu cần in cho $a0
```

```
syscall             # in ra màn hình
```

- Kết quả:

The screenshot displays a debugger interface with two main panels. The top panel, titled 'Data Segment', shows a memory dump starting at address 0x10010000. The data is organized into columns for different value sizes: Value (+0), Value (+4), Value (+8), Value (+c), Value (+10), Value (+14), Value (+18), and Value (+1c). The first row of data shows the string 'y u g N M n e h n a g n u T o N v 6 K 7' in ASCII. The bottom panel, titled 'Mars Messages', shows the output of the program. It includes a 'Reset: reset completed.' message, followed by the string 'Nguyen_Manh_Tung_VN07_K67' which was loaded into register \$a0. The final message is '-- program is finished running (dropped off bottom) --'. A 'Clear' button is visible next to the output messages.

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	y u g N	M n e	h n a	g n u T	o N v	6 K 7	\0 \0 \0 7	\0 \0 \0 \0
0x10010020	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010040	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010060	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010080	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x100100a0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x100100c0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x100100e0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010100	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010120	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0
0x10010140	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0	\0 \0 \0 \0

Mars Messages Run I/O

Reset: reset completed.

Nguyen_Manh_Tung_VN07_K67

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

Clear

+ Dữ liệu được cập nhật theo từng cụm 32-bit, theo chiều từ phải qua trái.

➔ Kết quả đúng với lí thuyết.

Assignment 2:

- Code:

```
.data

msg_1: .ascii "The sum of "
msg_2: .ascii " and "
msg_3: .ascii " is "

.text

add $s0, $zero, 28      # $s0=28
add $s1, $zero, 10      # $s1=10
add $t0, $s0, $s1       # $t0=$s0+$s1

# print msg_1
li $v0, 4
la $a0, msg_1
syscall

# print value of $s0
li $v0, 1
move $a0, $s0
syscall

# print msg_2
li $v0, 4
la $a0, msg_2
syscall

# print value of $s1
```

```

li $v0, 1

move $a0, $s1

syscall

# print msg_3

li $v0, 4

la $a0, msg_3

syscall

# print réult

li $v0, 1

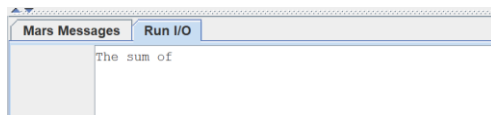
move $a0, $t0

syscall

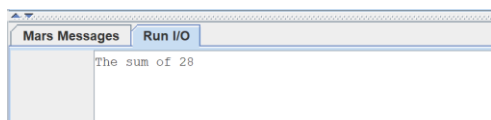
```

- Kết quả:

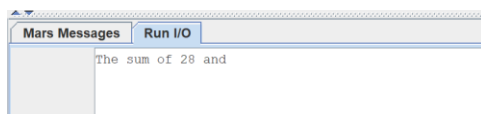
+ msg_1:



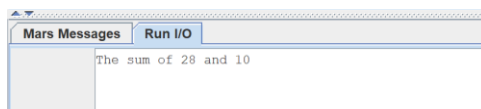
+ value of \$s0:



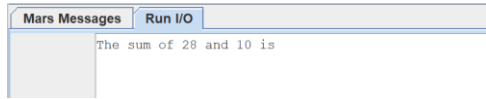
+ msg_2:



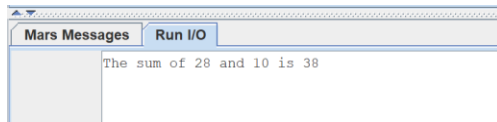
+ value of \$s1:



+ msg_3:



+ value:



➔ Kết quả đúng với lí thuyết.

Assignment 3:

- Code:

.data

x: .space 32 *# destination string x, empty*

y: .asciiz "Hello\n" *# source string y*

.text

strcpy:

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

la \$a1, y *# load adress of y to \$a1*

la \$a0, x *# load adress 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:
# print y
li $v0, 4
la $a0, y
syscall

# print x
li $v0, 4
la $a0, x
syscall

```

- Để copy chuỗi, thực hiện vòng lặp để copy từng kí tự của chuỗi y qua chuỗi x theo thứ tự từ trái qua phải. Kết quả được lưu giống như trong Assignment 1.

- Kết quả:

Data Segment			
Address	Value (+0)	Value (+4)	
0x10010000	l l e H	\0 \0 \n o	
0x10010020	l l e H	\0 \0 \n o	

=>

```

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

```

➔ Kết quả đúng với lí thuyết.

Assignment 4:

- Code:

```
.data

string: .space 50

msg_1: .ascii "Nhap xau: "
msg_2: .ascii "Do dai xau la: "

.text

main:

get_string:

    li $v0, 54                # get str fr dialog
    la $a0, msg_1            # load address msg_1 to $a0
    la $a1, string           # load address string to $s1
    la $a2, 50               # max 50 characters
    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
```

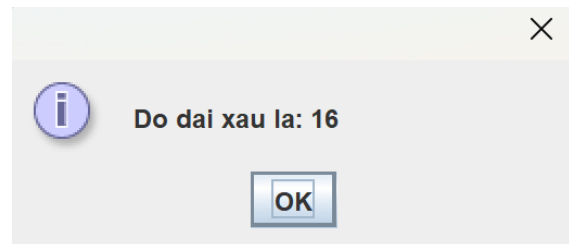
```

j check_char
end_of_str:
end_of_get_length:
print_length:
addi $t0, $t0, -1
li $v0, 56
la $a0, msg_2
move $a1, $t0
syscall

```

- Kết quả:

+ Kiểm tra độ dài xâu: “Nguyen_Manh_Tung” (length = 16)



➔ Kết quả đúng với lí thuyết.

Assignment 5:

- Code:

```

.data
get: .space 20
msg_1: .asciiz "Nhap ki tu thu "
msg_2: .asciiz ": "
msg_3: .asciiz "Ket qua la: "

```

msg_4: .asciiz "\n"

.text

li \$s0, 20 # max 20 characters

li \$s1, 0 # i=0

la \$s2, get # load address get to \$s2

li \$s3, 10

load_char:

beq \$s1, \$s0, end # i=20, exit

print msg_1

li \$v0, 4

la \$a0, msg_1

syscall

print ki tu thu i

addi \$t1, \$s1, 1

li \$v0, 1

move \$a0, \$t1

syscall

print msg_2

li \$v0, 4

la \$a0, msg_2

syscall

li \$v0, 12 # read char

syscall


```

move $t0, $v0
beq $t0, $s3, end # $t0="enter" -> exit
li $v0, 4
la $a0, msg_4
syscall
add $s5, $s2, $s1
sb $t0, 0($s5)
addi $s1, $s1, 1 # i=i+1
j load_char

end:
li $v0, 4
la $a0, msg_3
syscall

print:
li $v0, 11
lb $a0, 0($s5)
syscall
beq $s5, $s2, exit
addi $s5, $s5, -1
j print

exit:
li $v0, 10
syscall

```

- Kết quả:

+ TH1: ≤ 20 kí tự

```
Nhap ky tu thu 1: 1
Nhap ky tu thu 2: 2
Nhap ky tu thu 3: 3
Nhap ky tu thu 4: 4
Nhap ky tu thu 5: 5
Nhap ky tu thu 6:
Chuoi ky tu vua nhap (Bi dao nguoc thu tu) la: 54321
-- program is finished running --
```

+ TH2: > 20 kí tự

```
Nhap ki tu thu 11: sh
Nhap ki tu thu 12: dg
Nhap ki tu thu 13: j
Nhap ki tu thu 14: a
Nhap ki tu thu 15: h
Nhap ki tu thu 16: s
Nhap ki tu thu 17: v
Nhap ki tu thu 18: a
Nhap ki tu thu 19: s
Nhap ki tu thu 20: c
Ket qua la: csavshajdsajgdhsadãa
-- program is finished running --
```

=> chỉ cho phép nhập đến 20 kí tự rồi in ra kết quả

➔ Kết quả đúng với lí thuyết.