**REPORT WEEK 5**

Họ và tên: Nguyễn Văn Hưng

MSSV: 20225634

Lớp: Thực hành kiến trúc máy tính – Mã lớp: 147799

1. **Assignment 1:**
2. Đề bài: 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.
3. Code:

.data

test: .asciiz "Nguyen Van Hung 20225634"

.text

li $v0, 4 # $v0 = 4

la $a0, test # Dia chi cua test duoc ghi vao $a0

syscall # Loi goi dich vu he thong

1. Text segment and data segment: A screenshot of a computer

   Description automatically generated
2. Giải thích:

Đoạn mã này sử dụng hệ thống để in ra một chuỗi ký tự từ vùng nhớ dữ liệu. Dưới đây là phân tích chi tiết của nó:

* **.data**: Định nghĩa phần dữ liệu của chương trình.
* **test: .asciiz "Nguyen Van Hung 20225634"**: Khai báo một chuỗi ký tự có tên là **test** với nội dung là "Nguyen Van Hung 20225634". Chú ý rằng chuỗi này được kết thúc bằng ký tự null (asciiz).
* **.text**: Bắt đầu phần mã chương trình.
* **li $v0, 4**: Load immediate. Gán giá trị 4 vào thanh ghi $v0, đây là mã hợp lệ để gọi hệ thống để in ra chuỗi ký tự.
* **la $a0, test**: Lấy địa chỉ của chuỗi ký tự **test** và ghi vào thanh ghi $a0. Chuỗi ký tự này sẽ được in ra.
* **syscall**: Gọi hệ thống để thực hiện chức năng in ra chuỗi ký tự mà địa chỉ được lưu trong $a0.

Do mã hợp lệ trong $v0 là 4, lệnh **syscall** sẽ in ra chuỗi ký tự mà địa chỉ được lưu trong $a0.

1. Kết quả:

A screenshot of a computer

Description automatically generated

1. **Assignment 2:**
2. Đề bài: 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)”
3. Code:

.data

str1: .asciiz "The sum of "

str2: .asciiz " and "

str3: .asciiz " is "

.text

li $s0, 4 # number1 = 4

li $s1, 8 # number2 = 8

add $t0, $s0, $s1 # $t0 = Sum of 4 and 8

# Print string "str1"

li $v0, 4

la $a0, str1

syscall

# Print $s0

li $v0, 1

move $a0, $s0

syscall

# Print string "str2"

li $v0, 4

la $a0, str2

syscall

# Print $s1

li $v0, 1

move $a0, $s1

syscall

# Print string "str3"

li $v0, 4

la $a0, str3

syscall

# Print $t0

li $v0, 1

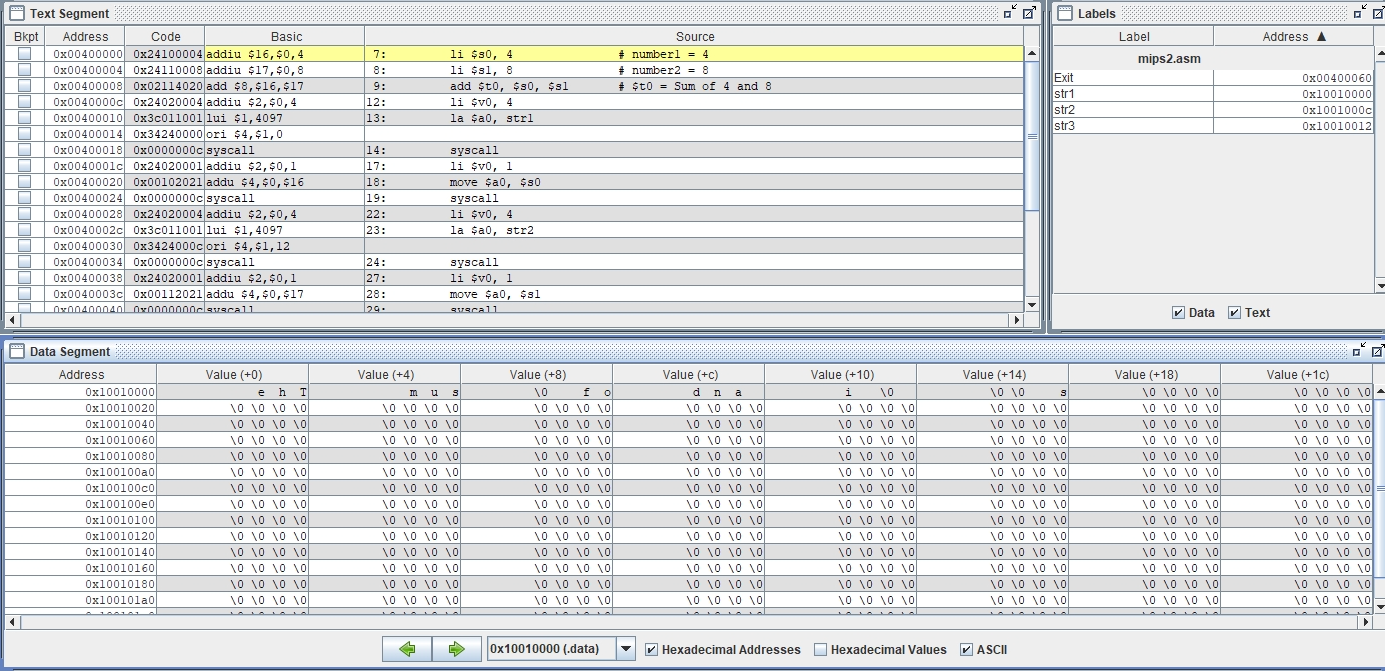
move $a0, $t0

syscall

Exit: li $v0, 10

Syscall

1. Text segment and data segment:



1. Giải thích

Đoạn mã MIPS này thực hiện việc tính tổng của hai số và in ra kết quả cùng với các thông điệp tương ứng. Dưới đây là phân tích chi tiết của đoạn mã:

* **.data**: Phần dữ liệu, nơi lưu trữ các chuỗi ký tự.
* **str1: .asciiz "The sum of "**: Chuỗi ký tự "The sum of " được lưu tại vị trí nhãn **str1**.
* **str2: .asciiz " and "**: Chuỗi ký tự " and " được lưu tại vị trí nhãn **str2**.
* **str3: .asciiz " is "**: Chuỗi ký tự " is " được lưu tại vị trí nhãn **str3**.
* **.text**: Phần mã chương trình.
* **li $s0, 4**: Gán giá trị 4 vào thanh ghi $s0, đây là số thứ nhất.
* **li $s1, 8**: Gán giá trị 8 vào thanh ghi $s1, đây là số thứ hai.
* **add $t0, $s0, $s1**: Thực hiện phép cộng giữa $s0 và $s1, kết quả được lưu vào thanh ghi $t0.
* Các lệnh từ **# Print string "str1"** đến **# Print $t0** in ra các thông điệp và giá trị tương ứng.
  + **li $v0, 4**: Gán giá trị 4 vào thanh ghi $v0 để thực hiện in chuỗi ký tự.
  + **la $a0, str1**: Đưa địa chỉ của chuỗi ký tự "The sum of " vào thanh ghi $a0.
  + **syscall**: Gọi hệ thống để in ra chuỗi ký tự.
    - Tương tự, các lệnh còn lại in ra số và các chuỗi ký tự " and " và " is ".
* **Exit:**: Nhãn kết thúc chương trình.
* **li $v0, 10**: Gán giá trị 10 vào thanh ghi $v0 để thực hiện kết thúc chương trình.
* **syscall**: Gọi hệ thống để kết thúc chương trình.

Top of Form

1. Kết quả

A screenshot of a computer

Description automatically generated

1. **Assignment 3:**
2. Đề bài: 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.
3. Code:

.data

x: .space 32 # destination string x, empty

y: .asciiz "Hello" # 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

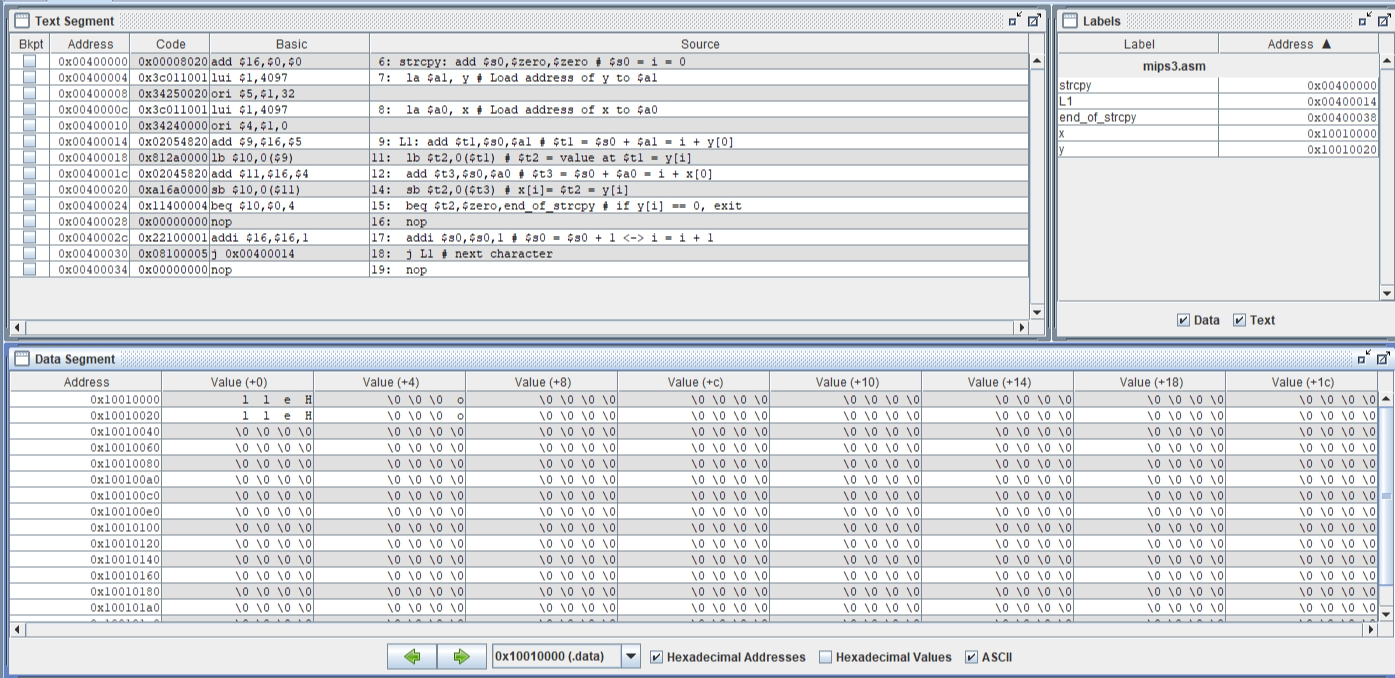
nop

addi $s0,$s0,1 # $s0 = $s0 + 1 <-> i = i + 1

j L1 # next character

nop

end\_of\_strcpy:

1. Text segment and data segment 
2. Giải thích
   * + Đoạn mã MIPS này thực hiện một hàm **strcpy** (copy string) để sao chép một chuỗi ký tự từ một vị trí nhớ đến một vị trí nhớ khác. Dưới đây là phân tích chi tiết của nó:
     + **.data**: Phần dữ liệu.
     + **x: .space 32**: Khai báo một vùng nhớ đủ lớn để lưu trữ chuỗi đích **x**. Trong trường hợp này, vùng nhớ được cấp là 32 byte.
     + **y: .asciiz "Hello"**: Chuỗi ký tự nguồn **y** được khai báo với nội dung là "Hello".
     + **.text**: Phần mã chương trình.
     + **strcpy:**: Nhãn định nghĩa hàm **strcpy**.
     + **add $s0,$zero,$zero**: Gán giá trị 0 vào thanh ghi $s0, đây là biến đếm i (index của chuỗi).
     + **la $a1, y**: Đưa địa chỉ của chuỗi nguồn **y** vào thanh ghi $a1.
     + **la $a0, x**: Đưa địa chỉ của chuỗi đích **x** vào thanh ghi $a0.
     + **L1:**: Nhãn bắt đầu vòng lặp.
     + **add $t1,$s0,$a1**: Tính địa chỉ của ký tự thứ **i** trong chuỗi nguồn **y** và lưu vào thanh ghi $t1.
     + **lb $t2,0($t1)**: Load byte từ địa chỉ được lưu trong $t1 (địa chỉ của ký tự thứ **i** trong chuỗi nguồn **y**) và lưu vào thanh ghi $t2.
     + **add $t3,$s0,$a0**: Tính địa chỉ của ký tự thứ **i** trong chuỗi đích **x** và lưu vào thanh ghi $t3.
     + **sb $t2,0($t3)**: Lưu byte từ thanh ghi $t2 (ký tự thứ **i** trong chuỗi nguồn **y**) vào địa chỉ được lưu trong $t3 (địa chỉ của ký tự thứ **i** trong chuỗi đích **x**).
     + **beq $t2,$zero,end\_of\_strcpy**: Kiểm tra nếu ký tự được sao chép là ký tự null (kết thúc chuỗi), thì nhảy đến nhãn **end\_of\_strcpy**.
     + **addi $s0,$s0,1**: Tăng giá trị của biến đếm i lên 1.
     + **j L1**: Nhảy đến nhãn **L1** để tiếp tục vòng lặp.
     + **end\_of\_strcpy:**: Nhãn kết thúc hàm **strcpy**.
3. Kết quả

A screenshot of a table

Description automatically generated

1. **Assignment 4 :**
2. Đề bài: Accomplish the Home Assignment 3 with syscall function to get a string from dialog and show the length to message dialog.
3. Code:

.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

move $a1, $t0

syscall

1. Text segment and data segment A screenshot of a computer

   Description automatically generated
2. Giải thích

Đoạn mã này thực hiện việc nhập một chuỗi từ người dùng, sau đó tính độ dài của chuỗi và in ra độ dài này. Dưới đây là phân tích chi tiết của nó:

* + **.data**: Phần dữ liệu.
  + **string: .space 50**: Khai báo một vùng nhớ có độ dài 50 byte để lưu trữ chuỗi nhập vào từ người dùng.
  + **Message1: .asciiz "Nhap xau: "**: Chuỗi thông điệp "Nhap xau: " được lưu trong vùng nhớ dữ liệu.
  + **Message2: .asciiz "Do dai xau la: "**: Chuỗi thông điệp "Do dai xau la: " được lưu trong vùng nhớ dữ liệu.
  + **.text**: Phần mã chương trình.
  + **main:**: Nhãn định nghĩa hàm chính **main**.
  + **get\_string:**: Nhãn định nghĩa hàm **get\_string**, để nhập chuỗi từ người dùng.
  + **li $v0, 54**: Load immediate. Gán giá trị 54 vào thanh ghi $v0, đây là mã hợp lệ để nhập một chuỗi từ hộp thoại.
  + **la $a0, Message1**: Đưa địa chỉ của chuỗi thông điệp "Nhap xau: " vào thanh ghi $a0.
  + **la $a1, string**: Đưa địa chỉ của vùng nhớ dành cho chuỗi **string** vào thanh ghi $a1.
  + **la $a2, 50**: Gán giá trị 50 vào thanh ghi $a2, đây là độ dài tối đa của chuỗi được nhập.
  + **syscall**: Gọi hệ thống để hiển thị hộp thoại nhập chuỗi và lưu chuỗi vào vùng nhớ **string**.
  + **get\_length:**: Nhãn định nghĩa hàm **get\_length**, để tính độ dài của chuỗi vừa nhập.
  + **la $a0,string**: Đưa địa chỉ của chuỗi vào thanh ghi $a0.
  + **add $t0,$zero,$zero**: Gán giá trị 0 vào thanh ghi $t0, đây là biến đếm i (index của chuỗi).
  + **check\_char:**: Nhãn bắt đầu vòng lặp để kiểm tra từng ký tự của chuỗi.
  + **add $t1,$a0,$t0**: Tính địa chỉ của ký tự thứ **i** trong chuỗi và lưu vào thanh ghi $t1.
  + **lb $t2, 0($t1)**: Load byte từ địa chỉ được lưu trong $t1 và lưu vào thanh ghi $t2.
  + **beq $t2, $zero, end\_of\_str**: Kiểm tra nếu ký tự là ký tự null (kết thúc chuỗi) thì nhảy đến nhãn **end\_of\_str**.
  + **addi $t0, $t0, 1**: Tăng giá trị của biến đếm i lên 1.
  + **j check\_char**: Nhảy đến nhãn **check\_char** để kiểm tra ký tự tiếp theo trong chuỗi.
  + **end\_of\_str:**: Nhãn kết thúc chuỗi, đánh dấu kết thúc vòng lặp.
  + **end\_of\_get\_length:**: Nhãn kết thúc hàm **get\_length**.
  + **print\_length:**: Nhãn định nghĩa hàm **print\_length**, để in ra độ dài của chuỗi.
  + **addi $t0, $t0, -1**: Giảm giá trị của biến đếm i đi 1, vì sau khi kiểm tra ký tự null, biến đếm sẽ tăng lên thêm 1.
  + **li $v0, 56**: Load immediate. Gán giá trị 56 vào thanh ghi $v0, đây là mã hợp lệ để in ra một giá trị nguyên.
  + **la $a0, Message2**: Đưa địa chỉ của chuỗi thông điệp "Do dai xau la: " vào thanh ghi $a0.
  + **move $a1, $t0**: Đưa giá trị của biến đếm i (độ dài của chuỗi) vào thanh ghi $a1.
  + **syscall**: Gọi hệ thống để in ra độ dài của chuỗi.

1. Top of Form
2. Kết quả

Nếu ta nhập chuỗi “hung”

A screenshot of a computer

Description automatically generated

Kết quả trả về là 4 thì hoàn toàn đúng

A screenshot of a computer

Description automatically generated

1. **Assignment 5 :**
2. Đề bài: 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.
3. Code:

.data

get\_char: .space 20

message1: .asciiz "Nhap ky tu thu "

message2: .asciiz ": "

message3: .asciiz "\n"

message4: .asciiz "Chuoi ky tu vua nhap la: "

.text

li $s0, 20 # N = 20

li $s1, 0 # i = 0

la $s2, get\_char # 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

# Show message "Nhap ky tu thu i: "

li $v0, 4

la $a0, message1

syscall

addi $t1, $s1, 1

li $v0, 1

move $a0, $t1

syscall

li $v0, 4

la $a0, message2

syscall

li $v0, 12 # Read character

syscall

move $t0, $v0

beq $t0, $s3, end\_read\_char # Press "Enter" branch to exit

li $v0, 4

la $a0, message3

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, message4

syscall

print\_string:

li $v0, 11

lb $a0, 0($s5)

syscall

beq $s5, $s2, exit

addi $s5, $s5, -1

j print\_string

exit:

li $v0, 10

syscall

1. Text segment and data segment A screenshot of a computer

   Description automatically generated
2. Giải thích
   * + - Đoạn mã này thực hiện việc nhập và in ra một chuỗi ký tự từ người dùng. Dưới đây là phân tích chi tiết của nó:
   * **.data**: Phần dữ liệu.
     + **get\_char: .space 20**: Khai báo một vùng nhớ có độ dài 20 byte để lưu trữ chuỗi ký tự.
     + **message1: .asciiz "Nhap ky tu thu "**: Chuỗi thông điệp "Nhap ky tu thu ".
     + **message2: .asciiz ": "**: Chuỗi thông điệp ": ".
     + **message3: .asciiz "\n"**: Chuỗi thông điệp "\n" để xuống dòng.
     + **message4: .asciiz "Chuoi ky tu vua nhap la: "**: Chuỗi thông điệp "Chuoi ky tu vua nhap la: ".
   * **.text**: Phần mã chương trình.
     + **li $s0, 20**: Gán giá trị 20 vào thanh ghi $s0, đây là số lượng ký tự tối đa có thể nhập.
     + **li $s1, 0**: Gán giá trị 0 vào thanh ghi $s1, đây là biến đếm i (index của chuỗi).
     + **la $s2, get\_char**: Đưa địa chỉ của vùng nhớ lưu trữ chuỗi **get\_char** vào thanh ghi $s2.
     + **li $s3, 10**: Gán giá trị 10 vào thanh ghi $s3, tương ứng với ký tự xuống dòng.
     + **read\_char:**: Nhãn định nghĩa hàm **read\_char**, để đọc ký tự từ người dùng.
       - **beq $s1, $s0, end\_read\_char**: Kiểm tra nếu biến đếm i bằng số lượng ký tự tối đa, thì nhảy đến nhãn **end\_read\_char** để kết thúc quá trình nhập.
       - Các lệnh từ **# Show message "Nhap ky tu thu i: "** đến **# Press "Enter" branch to exit** hiển thị thông điệp và đọc ký tự từ người dùng.
       - **add $s5, $s2, $s1**: Tính địa chỉ của ký tự thứ **i** trong chuỗi và lưu vào thanh ghi $s5.
       - **sb $t0, 0($s5)**: Lưu ký tự vừa đọc vào vị trí tương ứng trong chuỗi **get\_char**.
       - **addi $s1, $s1, 1**: Tăng giá trị của biến đếm i lên 1.
       - **j read\_char**: Nhảy đến nhãn **read\_char** để tiếp tục quá trình nhập.
     + **end\_read\_char:**: Nhãn kết thúc quá trình nhập chuỗi.
     + **print\_string:**: Nhãn định nghĩa hàm **print\_string**, để in ra chuỗi ký tự vừa nhập.
       - **li $v0, 11**: Load immediate. Gán giá trị 11 vào thanh ghi $v0, đây là mã hợp lệ để in ra một ký tự.
       - **lb $a0, 0($s5)**: Load byte từ địa chỉ được lưu trong $s5 (địa chỉ của ký tự thứ **i** trong chuỗi) và lưu vào thanh ghi $a0.
       - **beq $s5, $s2, exit**: Nếu địa chỉ hiện tại là địa chỉ đầu tiên của chuỗi, thì nhảy đến nhãn **exit** để kết thúc in chuỗi.
       - **addi $s5, $s5, -1**: Giảm địa chỉ hiện tại xuống 1, để trỏ tới ký tự trước đó trong chuỗi.
       - **j print\_string**: Nhảy đến nhãn **print\_string** để tiếp tục in ra các ký tự trong chuỗi.
     + **exit:**: Nhãn kết thúc chương trình.
       - **li $v0, 10**: Load immediate. Gán giá trị 10 vào thanh ghi $v0, đây là mã hợp lệ để kết thúc chương trình.
       - **syscall**: Gọi hệ thống để kết thúc chương trình.
3. Kết quả: A white background with black text

   Description automatically generated