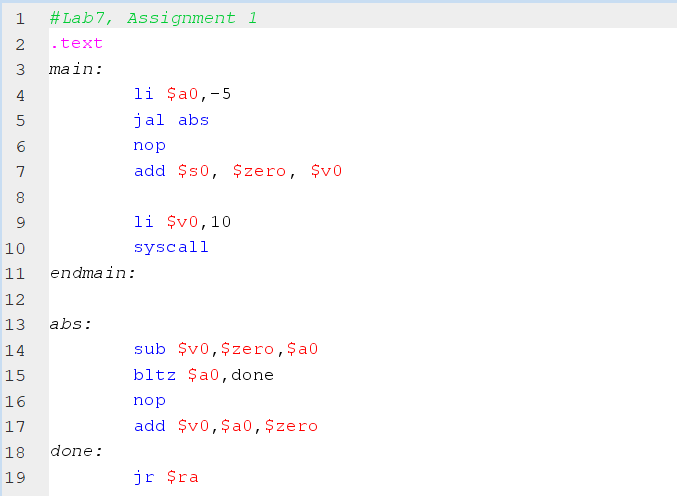
**Bài thực hành số 7**

**Lớp: 139365 – Học phần: Thực Hành Kiến Trúc Máy Tính   
Đào Minh Nhật – 20215107**

**Assignment 1:**

**Code:**

****

**Giải thích:**

Dòng 4: a0 là giá trị đầu vào

Dòng 5 : Thực hiện gọi tới hàm abs và đồng thời lưu trữ địa chỉ của lệnh tiếp theo (dòng số 6) vào thanh $ra.

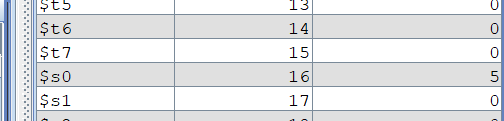
Ảnh có chứa bàn

Mô tả được tạo tự động

Ảnh có chứa Hình chữ nhật

Mô tả được tạo tự động

Dòng 7 : Gán giá trị của thanh ghi $v0 lúc này chứa kết quả cho thanh $s0. Lúc này thanh ghi $s0 là thanh chứa kết quả của hàm abs.



Dòng 9 10 : Kết thúc chương trình.

Dòng 14 : v0 = -a0

Dòng 15 : Nếu mà a0 < 0 thì jum tới done, lúc này thanh ghi $v0 chứa kết quả.

Dòng 17 : Ngược lại trường hợp a0 > 0 thì gán luôn giá trị của a0 vào thanh ghi $v0.

Dòng 19 : Thực hiện jum tới địa chỉ đã được lưu trong thanh $ra là dòng 6.

**Assignment 2**

**Code :**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

**Giải thích :**

Dòng 3-5 : Nhập 3 số nguyên

Dòng 6: Thực hiện gọi tới hàm max và đồng thời lưu trữ địa chỉ của lệnh tiếp theo (dòng số 7) vào thanh $ra

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa bàn

Mô tả được tạo tự động

Dòng 10: v0 = a0 ( v0 là thanh ghi lưu trữ số lớn nhất. được khởi tạo bằng số đầu tiên a0 ).

Dòng 11: t0 = a1 – v0

Dòng 12: Nếu t0 < 0 tức là a1 < v0 thì gọi tới hàm okay ngược lại thì thực hiện tiếp dòng 13

Dòng 14: v0 = a1 => cập nhật lại giá trị lớn nhât.

Dòng 16: t0 = a2 – v0

Dòng 17: Nếu t0 < 0 tức là a2 < v0 thì thực hiện hàm done (đã tìm được max)

Dòng 19: Cập nhật lại giá trị max v0 = a2

Dòng 20: Thực hiện lệnh mà địa chỉ đã được lưu ở trong thanh ghi $ra.

**Kết quả:**

**Ảnh có chứa bàn

Mô tả được tạo tự động**

**Assignment 3**

**Code:**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

**Giải thích:**

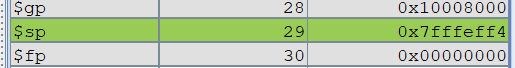
Dòng 3-4: Khởi tạo s0, s1



Dòng 5: Giảm địa chỉ của ngăn xếp $sp đi 8byte để mở rộng ngăn xếp để có thể lưu trữ s0 và s1.

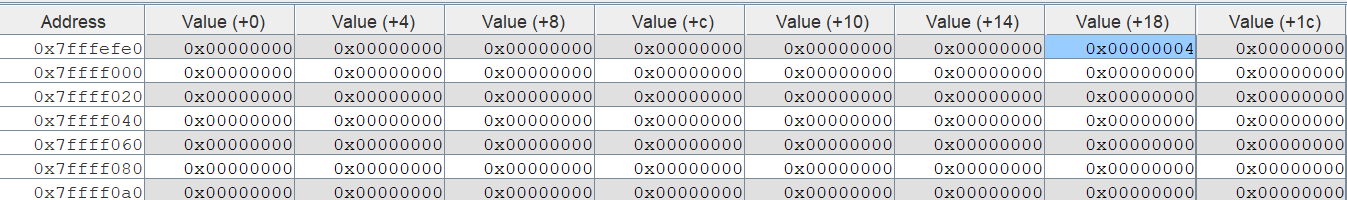
Trước: Ảnh có chứa văn bản

Mô tả được tạo tự động

Sau: 

Dòng 6-7: Lần lượt đẩy s0 và s1 vào trong ngăn xếp.

+ Giá trị $s0 được lưu trữ tại địa chỉ 4($sp)



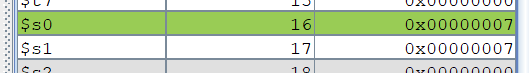
+ Giá trị $s1 được lưu trữ tại địa chỉ 0($sp)

Ảnh có chứa bàn

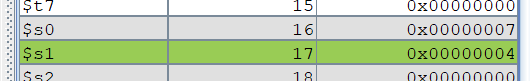
Mô tả được tạo tự động

Dòng 11-12: Lần lượt lấy giá trị ở trong ngăn xếp rồi gán vào thanh ghi $s0, $s1.

+ Giá trị $s0 được lấy từ địa chỉ 0($sp) và được lưu trữ lại trong $s0.



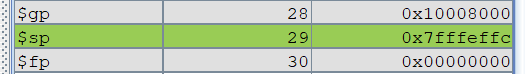
+ Giá trị $s1 được lấy từ địa chỉ 4($sp) và được lưu trữ lại trong $s1.



Dòng 13: Con trỏ ngăn xếp $sp được điều chỉnh trở lại vị trí ban đầu bằng cách giảm địa chỉ ngăn xếp đi 8 byte và trả lại kích thước ban đầu cho ngăn xếp.

Trước: Ảnh có chứa bàn

Mô tả được tạo tự động

Sau: 

**Assignment 4**

**Code:**

**Ảnh có chứa văn bản

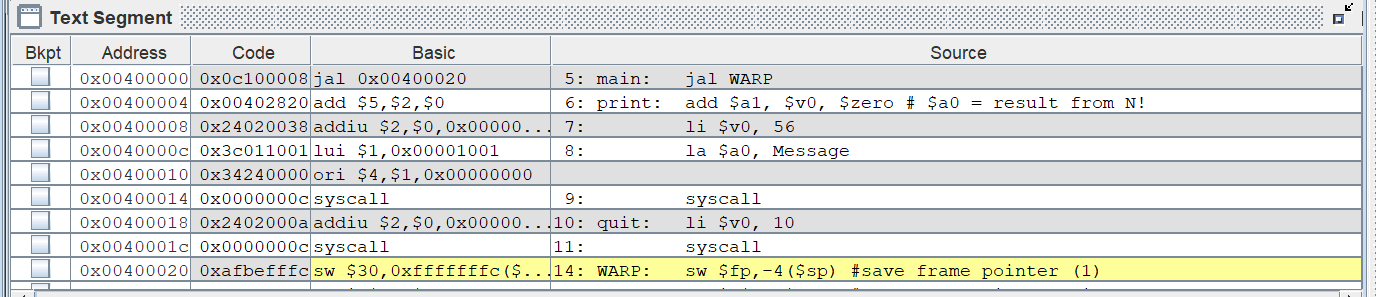
Mô tả được tạo tự động**

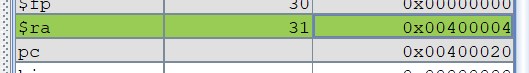
**Ảnh có chứa văn bản

Mô tả được tạo tự động**

**Giải thích:**

Dòng 5: Thực hiện hàm WARP (Dòng 14) đồng thời lưu trữ địa chỉ của lệnh tiếp theo (Dòng 6) vào thanh ghi $ra = 0x00400004





ĐỊA CHỈ STACK $sp = 0x7fffeffc (1)

Dòng 14: Giá trị $fp (frame pointer) được lưu trữ tại địa chỉ -4($sp)

Thanh ghi $sp: 

Nơi giá trị của thanh $fp được lưu: Ảnh có chứa bàn

Mô tả được tạo tự động

Dòng 15: Tạo frame pointer mới trỏ tới đỉnh của stack

Ảnh có chứa văn bản

Mô tả được tạo tự động

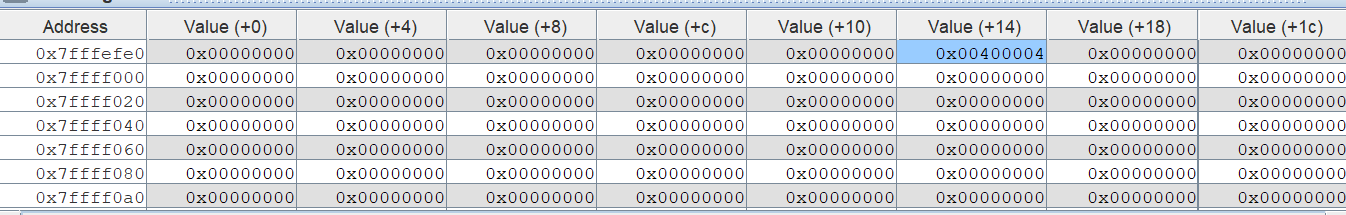
Dòng 16: Thay đổi con trỏ trỏ tới stack:

Ảnh có chứa văn bản

Mô tả được tạo tự động

ĐỊA CHỈ STACK $sp = 0x7fffeff4 (2)

Dòng 17: Lưu trữ địa chỉ trả về (địa chỉ của lệnh printf) tại 0($sp)

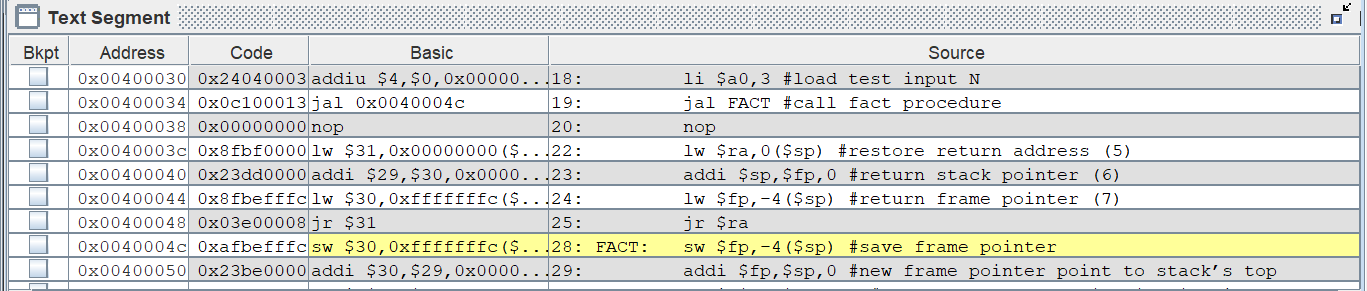


Dòng 18: Khởi tạo giá trị cần tính tại $a0. (a0 = 3)

Dòng 19: Gọi tới hàm FACT ( Dòng 28) để cập nhật các giá trị vào stack. Đồng thời lưu trữ lệnh tiếp theo ( Dòng 20 ) vào $ra

Ảnh có chứa văn bản

Mô tả được tạo tự động



Dòng 28: Lưu trữ $fp (frame pointer) vào địa chỉ -4($sp)

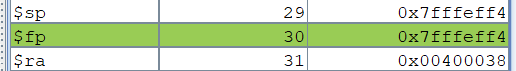
$fp = 0x7fffeffc

Ảnh có chứa bàn

Mô tả được tạo tự động

Dòng 29: Tạo frame pointer mới trỏ tới đỉnh của stack

$fp = $sp = 0x7fffeff4



Dòng 30: Cấp cho stack 12byte dùng để lưu trữ $fp,$ra,$a0

Ảnh có chứa bàn

Mô tả được tạo tự động

ĐỊA CHỈ STACK $sp = 0x7fffefe8 (3)

Dòng 31: Lưu trữ địa chỉ trả về dòng 20 tại 4($sp)Ảnh có chứa bàn

Mô tả được tạo tự động

Dòng 32: Lưu trữ giá trị a0 = 3 tại 0($sp).

Ảnh có chứa bàn

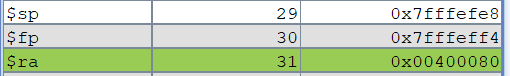
Mô tả được tạo tự động

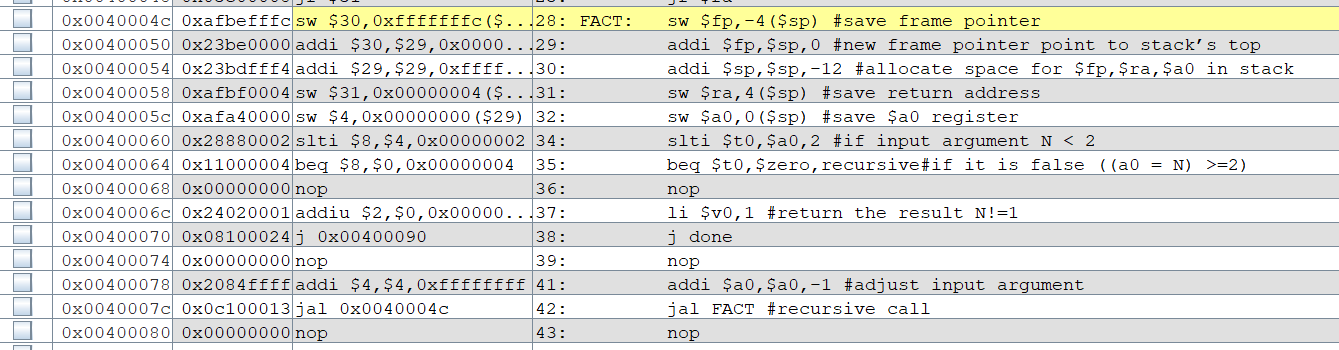
Dòng 34-35: Kiểm tra nếu a0 > = 2 thì thực hiện hàm recursive.

a0 = 3 => thực hiện hàm recursive.

Dòng 41: Giảm a0. a0 = a0 – 1 = 2.

Dòng 42: Gọi lại hàm FACT(dòng 28). Đồng thời lưu trữ địa chỉ trả về dòng 43 tại $ra





Dòng 28: lưu trữ $fb = 0x7fffeff4 vào địa chỉ -4($sp).Ảnh có chứa bàn

Mô tả được tạo tự động

Dòng 29: Tạo frame pointer mới trỏ tới đỉnh của stack

$fp = $sp = 0x7fffefe8

Ảnh có chứa văn bản

Mô tả được tạo tự động

Dòng 30: Cấp cho stack 12byte dùng để lưu trữ $fp,$ra,$a0.

Ảnh có chứa bàn

Mô tả được tạo tự động

ĐỊA CHỈ STACK $sp = 0x7fffefdc (4)

Dòng 31-32: Lưu trữ địa chỉ trả về dòng 43 tại 4($sp) và lưu trữ giá trị a0 = 2 tại 0($sp).

Ảnh có chứa bàn

Mô tả được tạo tự động

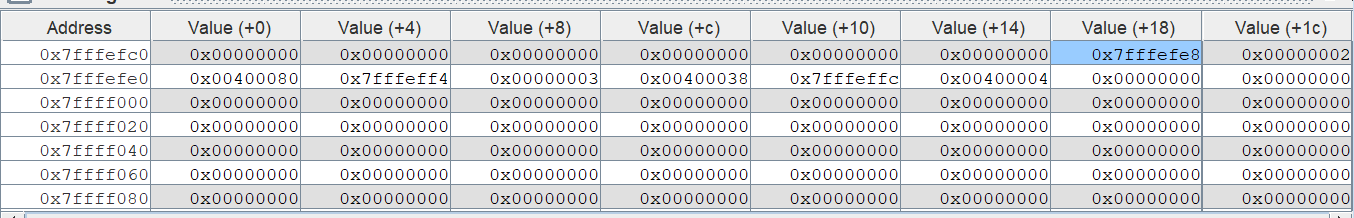
Dòng 34-35: Kiểm tra nếu a0 > = 2 thì thực hiện hàm recursive.

a0 = 2 => thực hiện hàm recursive.

Dòng 41: Giảm a0. a0 = a0 – 1 = 1.

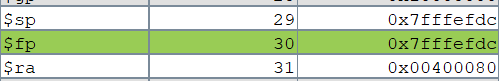
Dòng 42: Gọi lại hàm FACT(dòng 28). Đồng thời lưu trữ địa chỉ trả về dòng 43 tại $ra (tưởng tự trên).

Dòng 28: lưu trữ $fb = 0x7fffefe8 vào địa chỉ -4($sp).



Dòng 29: Tạo frame pointer mới trỏ tới đỉnh của stack

$fp = $sp = 0x7fffefdc



Dòng 30: Cấp cho stack 12byte dùng để lưu trữ $fp,$ra,$a0.

Ảnh có chứa văn bản

Mô tả được tạo tự động

ĐỊA CHỈ STACK $sp = 0x7fffefd0 (5)

Dòng 31-32: Lưu trữ địa chỉ trả về dòng 43 tại 4($sp) và lưu trữ giá trị a0 = 1 tại 0($sp) Ảnh có chứa bàn

Mô tả được tạo tự động

Dòng 34-35: Kiểm tra nếu a0 > = 2 thì thực hiện hàm recursive.

a0 = 1=> thực hiện dòng 36.

Dòng 37: Trả lại kết quả với a0 = 1 => v0 = 1! = 1

Dòng 38: Thực hiện hàm done.

Dòng 47-48: Lấy ra địa chỉ trả về dòng 43 $ra = 0x00400080 từ 4($sp) và giá trị a0 = 1 từ 0($sp).

Dòng 49: Khôi phục lại con trỏ stack trước đó là giá trị của $fp hiện tại.

Ảnh có chứa bàn

Mô tả được tạo tự động

Dòng 50: Khôi phục lại frame pointer được lưu tại -4($sp).

Ảnh có chứa bàn

Mô tả được tạo tự động

Dòng 51: Thực hiện jum tới địa chỉ đang được lưu trong thanh $ra. Hiện tại là địa chỉ của dòng 43.

Dòng 44: Load giá trị a0 = 2 đang lưu tại 0($sp) vào thanh $v1.

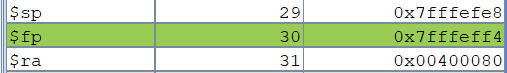
v1 = 2

Dòng 45-46: Nhân v1=2 với v0=1 rồi kết quả lưu vào thanh v0

Ảnh có chứa Hình chữ nhật

Mô tả được tạo tự động

Dòng 47-50: Khôi phục lại $ra = 0x00400080, $a0 = 2, $sp = 0x7fffefe8, $fp = 0x7fffeff4 được lấy ra từ stack.



Dòng 51: Thực hiện jum tới địa chỉ đang được lưu trong thanh $ra. Hiện tại là địa chỉ của dòng 43.

Dòng 44: Load giá trị a0 = 3 đang lưu tại 0($sp) vào thanh $v1.

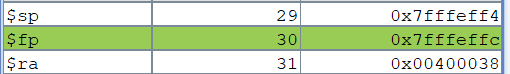
v1 = 3

Dòng 45-46: Nhân v1=3 với v0=2 rồi kết quả lưu vào thanh v0

Ảnh có chứa Hình chữ nhật

Mô tả được tạo tự động

Dòng 47-50: Khôi phục lại $ra = 0x00400038, $a0 = 3, $sp = 0x7fffeff4, $fp = 0x7fffeffc được lấy ra từ stack.

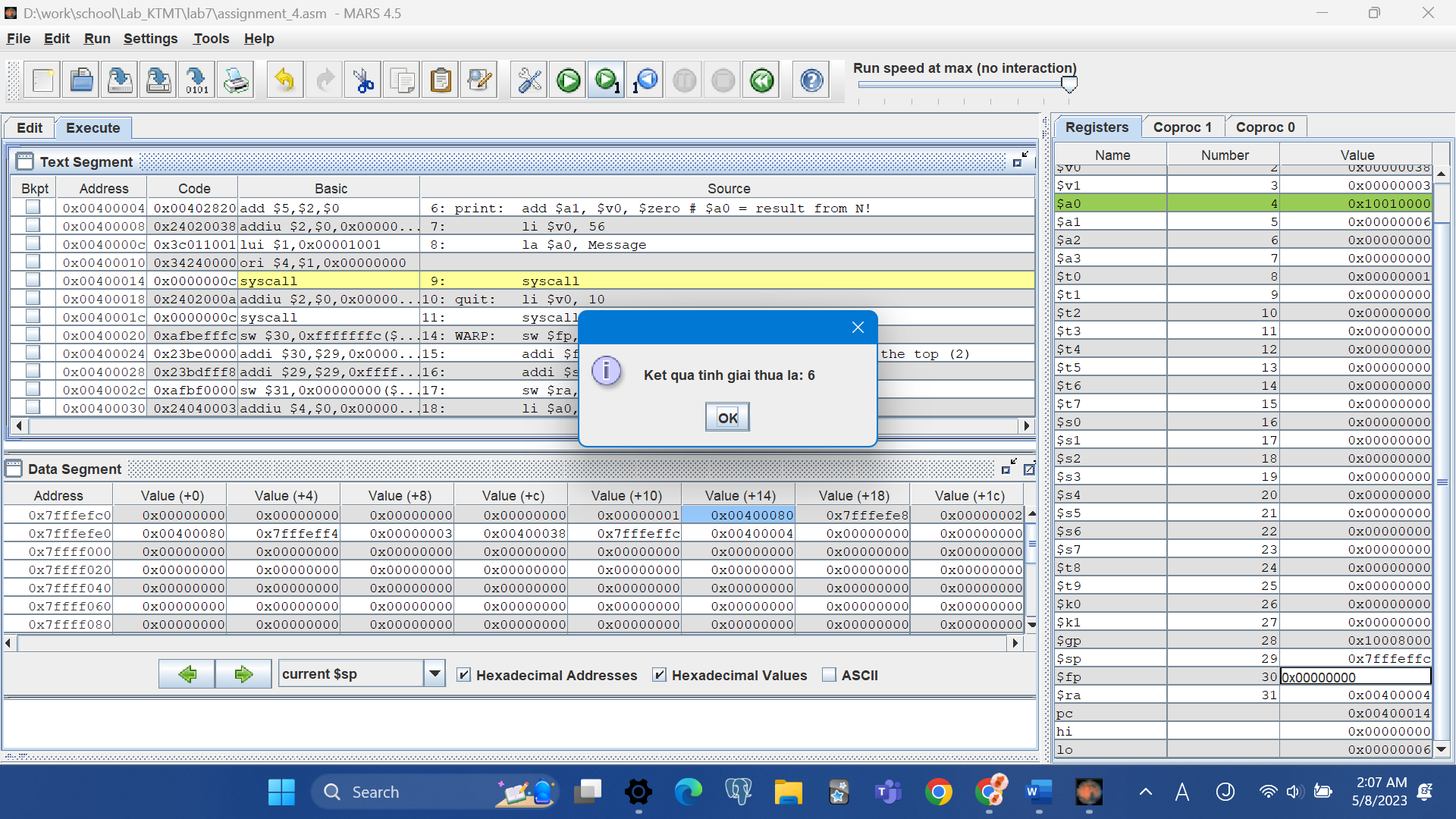


Dòng 51: Thực hiện jum tới địa chỉ đang được lưu trong thanh $ra. Hiện tại là địa chỉ của dòng 20.

Dòng 22-24: Khôi phục lại $ra = 0x00400004, $sp = 0x7fffeffc, $fp = 0x00000000

Dòng 25: Thực hiện jum tới địa chỉ đang được lưu trong thanh $ra. Hiện tại là địa chỉ của dòng 6.

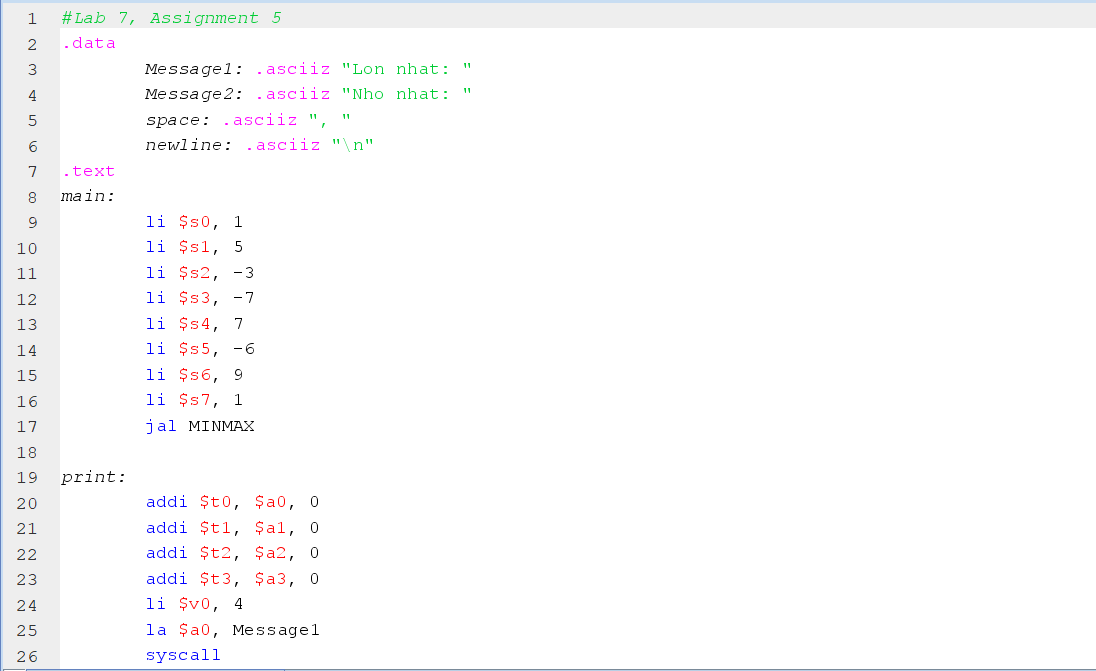
Dòng 6-9: Thực hiện in ra màn hình.

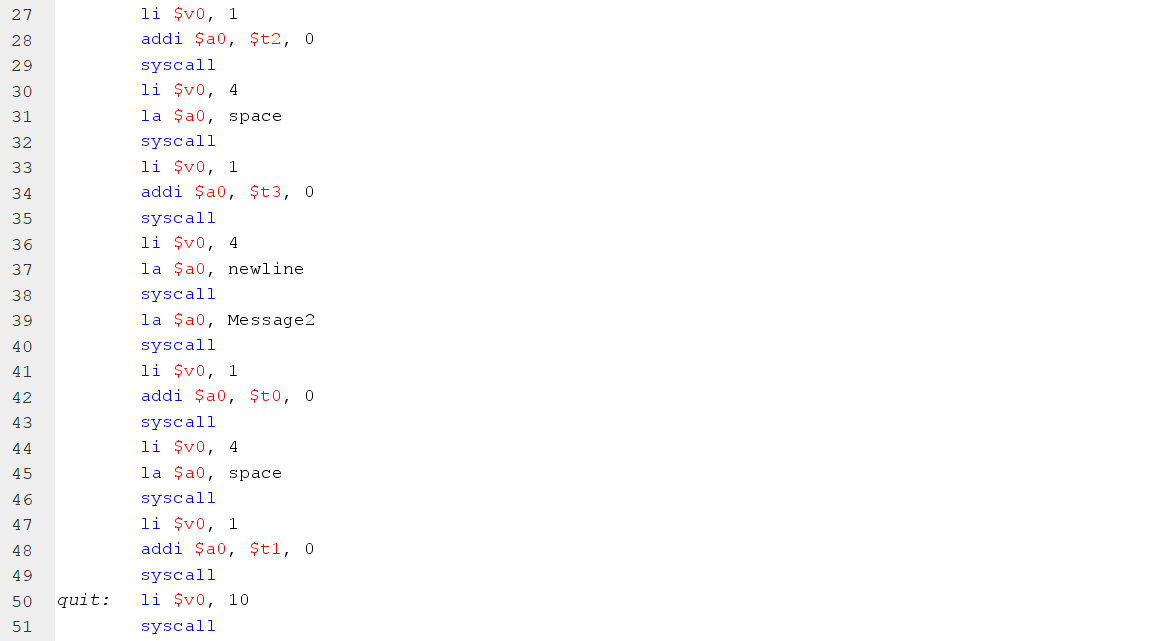


Dòng 10-11: Kết thúc chương trình.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  | $sp(3)-> | $ra(2) |
|  |  |  |  |  | $a0(2) = 1 |
|  |  |  |  |  | $fp(2) |
|  |  | $sp(2)-> | $ra(1) | $fp(3)->  $sp(2)>- | $ra(1) |
|  |  |  | $a0(1) = 2 |  | $a0(1) = 2 |
|  |  |  | $fp(1) |  | $fp(1) |
| $sp(1)-> | $ra(0) | $fp(2)->  $sp(1)-> | $ra(0) | $fp(2)->  $sp(1)-> | $ra(0) |
|  | $a0(0) = 3 |  | $a0(0) = 3 |  | $a0(0) = 3 |
|  | $fp(0) |  | $fp(0) |  | $fp(0) |
| $fp(1)->  $sp(0)-> |  | $fp(1)->  $sp(0)-> |  | $fp(1)->  $sp(0)-> |  |
|  | … |  | … |  | … |
| $fp(0)-> |  | $fp(0)-> |  | $fp(0)-> |  |

**Asignment 5**

**Code:**

****

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

**Giải thích:**

Dòng 9-16: Khai báo 8 phần tử

Dòng 17: Thực hiện hàm MINMAX (dòng 53), lưu địa chỉ lệnh tiếp theo vào thanh $ra.

Dòng 20-51: Thực hiện in ra kết quả và kết thúc chương trình.

Dòng 53: Bắt đầu hàm MINMAX

Dòng 54-56: Mở rộng ngăn xếp để lưu frame pointer ($fp) và địa chỉ trở về ($ra) đã được lưu trước đó.

Dòng 58-61: Khởi tạo giá trị và vị trí của số min, max.

Dòng 62: Mở rộng ngăn xếp để lưu trữ các phần tử.

Dòng 63-74: Lưu trữ các phần tử và các giá trị vửa khởi tạo của min, max vào ngăn xếp.

Dòng 76: Khởi tạo giá trị t0 để bắt đầu vòng lặp.

Dòng 77-82: Vòng lặp duyệt từ 0 đến 7 để tìm min, max.

Dòng 85-90: Lấy ra các giá trị, vị trí min max.

Dòng 91-96: Hàm tìm min. a0 là giá trị min tìm đc và a1 là vị trí

Dòng 97-102: Hàm tìm max. a2 là giá trị max tìm đc và a3 là vị trí.

Dòng 103-109: Kết thúc hàm so sánh compare. Đẩy kết quả vừa tìm được vào stack. Và trở lại thực hiện địa chỉ đã đc lưu trong stack để thực hiện lệnh tiếp theo.

Dòng 110-118: Kết thúc hàm MINMAX. Lấy ra các giá trị, vị trí vừa tìm đc vào trở về thực hiện lệnh tiếp theo để in kết quả ra màn hình.

**Kết quả:**

**Ảnh có chứa bàn

Mô tả được tạo tự động**

Với các giá trị như trên, ta thu được kết quả dưới đây sau khi thực hiện chương trình:

**Ảnh có chứa văn bản

Mô tả được tạo tự động**