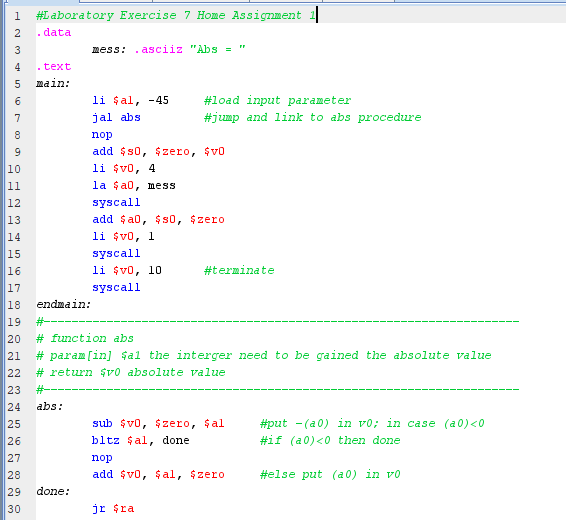
**Bài thực hành số 7**

**Lớp: 139365 – Học phần: Thực hành Kiến Trúc Máy Tính**

**Họ và tên: Đinh Thị Hồng Phúc MSSV: 20215118**

Bài 1.



*Thực hiện gõ chương trình vào công cụ* ***MARS***

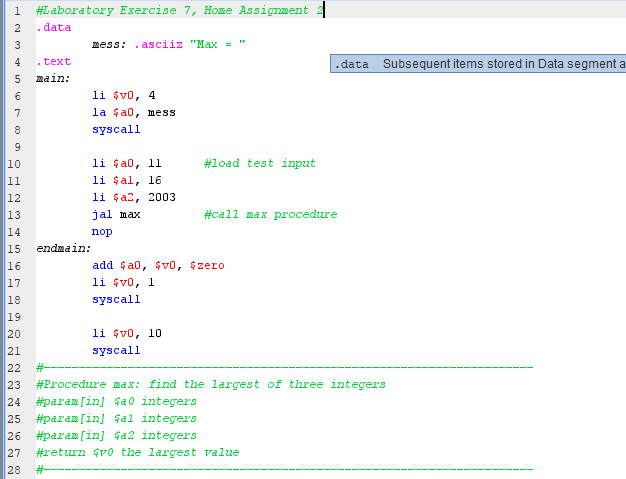
Giải thích:

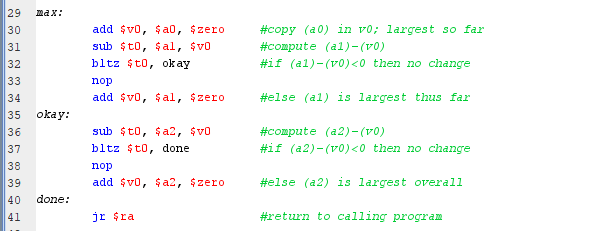
* Trong hàm *main*, khai báo giá trị cần tính trị tuyệt đối vào thanh ghi $a1. Lệnh *jal* để nhảy tới phần *abs* sau đó quay lại đây và sẽ lưu địa chỉ trả về của chương trình ở thanh ghi $ra. Dòng lệnh từ 10-17 là để in giá trị tuyệt đối ra màn hình và kết thúc chương trình. Giá trị tuyệt đối được lưu ở thanh ghi $s0 (dòng 9)
* Trong hàm *abs*, kiểm tra số có âm hay không bằng cách lấy 0 trừ đi số ban đầu, sau đó so sánh giá trị có được với 0. Nếu số ban đầu là số âm thì giá trị tuyệt đối của nó được lưu vào thanh ghi $v0 (dòng 28). Nếu là số dương thì chương trình in ra số ban đầu
* Dòng 30: Trở về địa chỉ được lưu trong thanh ghi $ra để tiếp tục chương trình



*Thực hiện chạy chương trình với* ***MARS***

Bài 2.





*Thực hiện gõ chương trình vào công cụ* ***MARS***

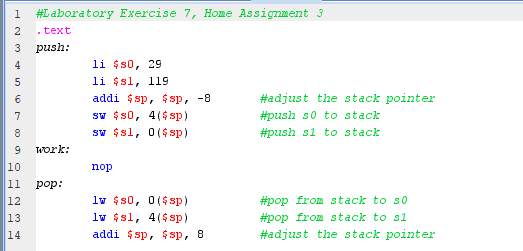
Giải thích:

* Trong hàm *main,* in chuỗi “Max = ” ra màn hình, khai báo 3 giá trị so sánh với nhau
* Hàm *end\_main*, in giá trị lớn nhất ra màn hình và kết thúc chương trình
* Hàm *max,* ban đầu giả sử giá trị lớn nhất là a0 và gán vào thanh ghi $v0. So sánh các số còn lại với v0 bằng các lấy số đó trừ đi v0, nếu được số nhỏ hơn 0 thì vẫn giữ nguyên giá trị v0, nếu lớn hơn 0 thì gán giá trị mới cho v0
* Giá trị lớn nhất được trả về bởi hàm *max* được lưu trong thanh ghi $v0



*Thực hiện chạy chương trình với* ***MARS***

Bài 3.



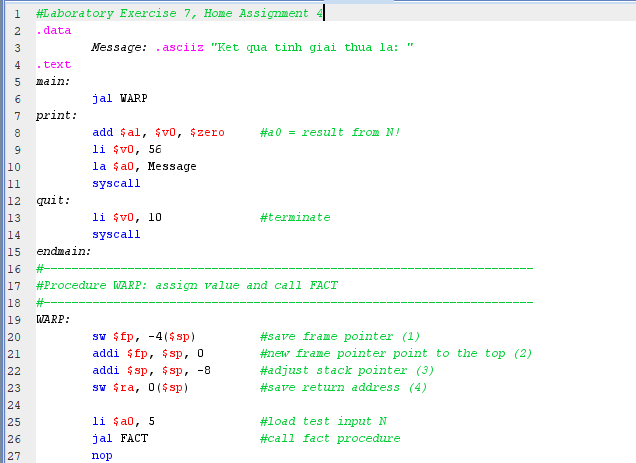
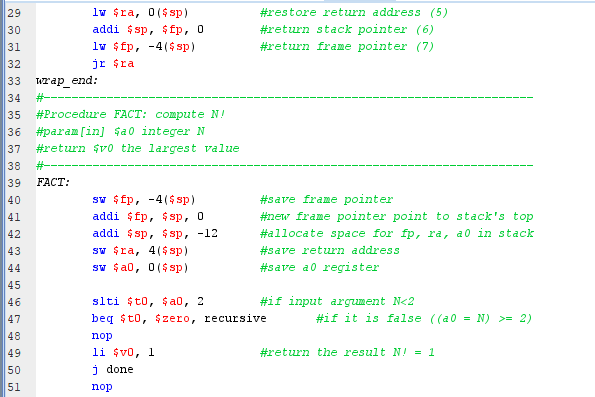
*Thực hiện gõ chương trình vào công cụ* ***MARS***

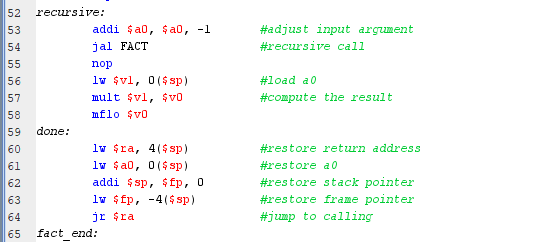
Giải thích:

* Hàm *push*, gán giá trị $s0, $s1 lần lượt là 29, 119. Sau đó điều chỉnh con trỏ stack bằng cách giảm giá trị của $sp đi 8 byte (push 2 giá trị vào stack). Lưu giá trị của $s0, $s1 vào stack bằng lệnh *sw*, địa chỉ được $s0, $s1 lưu trữ tại 4($sp) và 0($sp)
* Hàm *pop*, lấy giá trị từ stack bằng lệnh *lw*, lưu lần lượt các giá trị được lấy ra vào thanh ghi $s0, $s1. Sau đó điều chỉnh con trỏ stack về vị trí ban đầu bằng cách tăng giá trị $sp lên 8 byte (pop 2 giá trị)



*Thực hiện chạy chương trình với* ***MARS***

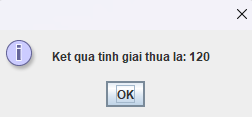
Bài 4.  



*Thực hiện gõ chương trình vào công cụ* ***MARS***

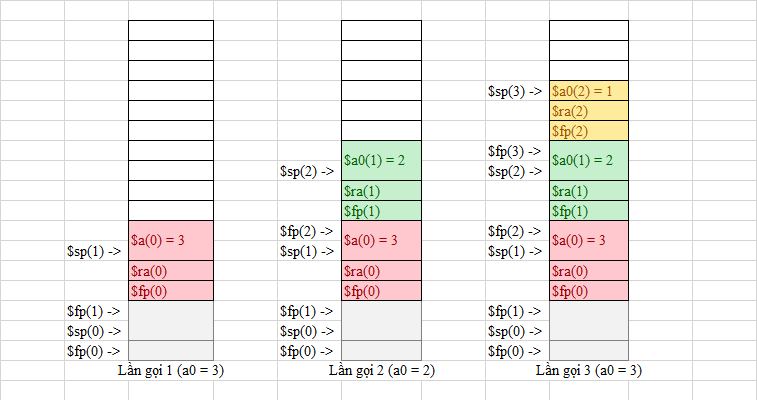
Giải thích:

* Dòng 7-14: In kết quả ra dialog và kết thúc chương trình
* Hàm *WARP*: Thực hiện các các bước để tính N! bằng cách gọi hàm *FACT*
* Hàm *FACT*: Tính giai thừa bằng cách sử dụng đệ qui. Nhận số nguyên N đầu vào $a0 và trả về giá trị giai thừa $v0. Kiểm tra điều kiện dừng khi N<2 (dòng 46) thì hàm trả về giá trị 1. Nếu N >=2 thì tính giai thừa của N-1 bằng cách gọi đến chính hàm *FACT* và nhân với N để được kết quả cuối cùng
* Các dòng lệnh 20-23, 29-32, 40-44, hàm *done* dùng để lưu trữ và lấy lại địa chỉ con trỏ của frame ban đầu, tạo con trỏ mới cho frame mới

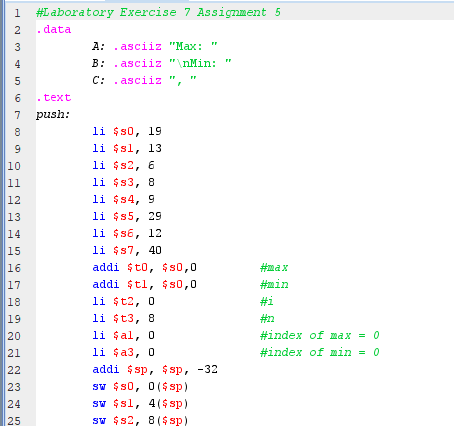


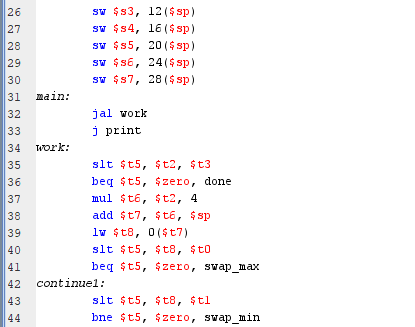
*Thực hiện chạy chương trình với* ***MARS***

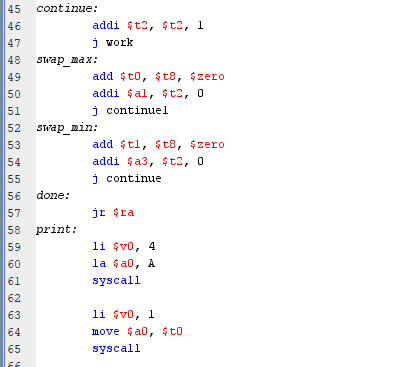
Vẽ stack:

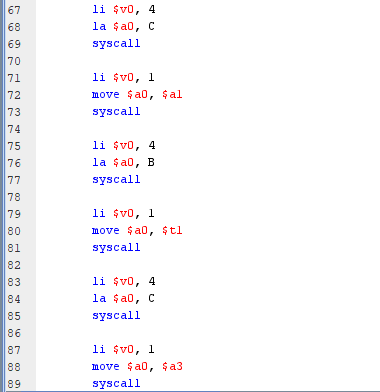


Bài 5.









*Thực hiện gõ chương trình vào công cụ* ***MARS***

Giải thích:

* Hàm *push:* Gán giá trị dãy số vào các thanh ghi từ $s0 đến $s8, sau đó đẩy các số vào stack. Khởi tạo các biến max, min, biến đếm, độ dài dãy số và vị trí của max, min
* Hàm *main*: Sử dụng hàm *work* để tìm giá trị lớn nhất và nhỏ nhất của dãy số, sau đó in ra màn hình các giá trị này bằng hàm *print*
* Hàm *work*: Tìm giá trị lớn nhất, nhỏ nhất bằng cách so sánh giá trị của $t2 với giá trị lớn nhất hiện tại, nếu lớn hơn thì sẽ cập nhật giá trị lớn nhất, chỉ số sẽ được lưu vào thanh ghi $a1. Nếu giá trị của $t2 nhỏ hơn giá trị nhỏ nhất hiện tại thì sẽ cập nhật giá trị nhỏ nhất, chỉ số sẽ được lưu vào thanh ghi $a3.



*Thực hiện chạy chương trình với* ***MARS***