**CHƯƠNG TRÌNH CON**

**(Read section 8.0 Page 67)**

**1. Procedure Format**

The general syntax is as follows:

**.globl procedureName**

**.ent procedureName**

**procedureName:**

*# code goes here*

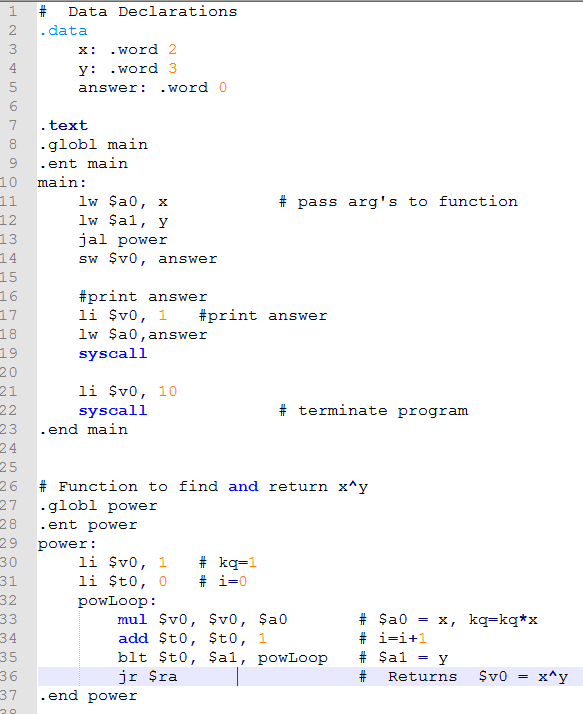
**.end procedureName**

* Dùng **jal** và **jr** (jal: jump and link; jr: jump return)

jal procedureName # Call procedureName

jr $ra # Return from procedure

* **VD: Viết chương trình tính lũy thừa x^y, trả về kết quả answer = power(x, y)**



**Bài tập thực hành:**

1/ Viết lại chương trình tính lũy thừa x^y, với x, y nhập từ bàn phím (Dùng chương trình con Power).

2^4=(2.2.2).2=((2.2).2).2

2^10=(2…2).2

2/ Viết chương trình con tìm số min/max trong 2 số x, y (với x, y nhập từ bàn phím).

2.1 Viết chương trình tìm số min/max của mảng A có n phần tử, nhập vào từ bàn phím.

3/ Viết chương trình con tính n! =1.2.3…n=(n-1)!n,

Y=1;n=5

For(i=1;I<=5;i++)

{

Y=Y.i;

}

i=1;y=1.1=1, 1!

I=2;y=1.2=2, 2!=1.2

I=3;y=2.3=6; 3!=1.2.3=6

I=4;y=6.4=24; 4!=1.2.3.4=24

2^3=2.2.2=8=(2.2).2=4.2

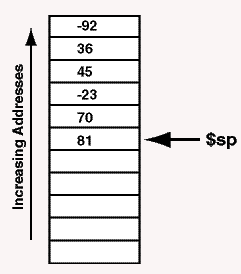
2^10= 2…2=1024=(2…2).2=512.2

**STACK**

(Read Section 7.0, Page 61)

**1. Stack:**

* Stack = LIFO (Last in - First out): Ngăn xếp (Stack) là vùng nhớ được truy cập theo cơ chế như băng đạn, nghĩa là phần tử “vào trước ra sau” hoặc “vào sau ra trước”.



* Mỗi phần tử trong bộ nhớ có kích thước một word (32 bit = 4 bytes).
* Thanh ghi **$sp** (stack pointer) là thanh ghi luôn luôn trỏ đến đỉnh của stack. Đỉnh của Stack luôn có địa chỉ thấp hơn.
* Stack có 2 thao tác cơ bản: **Push:** thêm 1 phần tử vào stack; **Pop:** Lấy 1 phần tử ra khỏi Stack

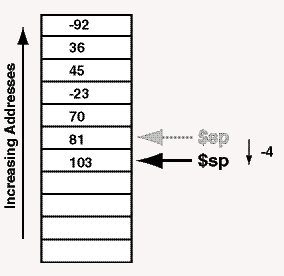
***VD1:* Giả sử đỉnh Stack đang trỏ vào số 81 như hình ở trên. Nếu bạn muốn thêm một phần tử 103 vào Stack thì giảm địa chỉ con trỏ $sp xuống 4 bytes, sau đó push giá trị vào Stack.**

*li $t0, 103 # load giá trị vào $t0*

*subu $sp, $sp, 4 # Giảm xuống 4 bytes để con trỏ* ***$sp*** *trỏ vào đỉnh Stack mới*

*sw $t0, ($sp) # push giá trị vào đỉnh Stack mới*

*(Xem minh họa như hình bên dưới)*

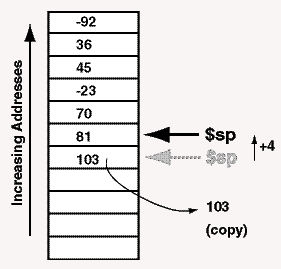


**VD2: Lấy giá trị trong đỉnh của Stack ra**

lw $t1, ($sp) # Lấy giá trị 103 trong đỉnh Stack, gán vào St1

addu $sp, $sp, 4 # tăng lên 4 bytes để con trỏ **$sp** trỏ vào đỉnh Stack mới

*(Xem minh họa như hình bên dưới)*



**VD 3: Viết Chương trình đọc các phần tử trong mảng đã cho và lưu vào stack; Lấy các phần tử trong stack lưu trở lại vào mảng; Xuất các phần tử trong mảng mới ra màn hình. (Chương trình này minh họa việc đảo ngược các phần tử bằng cách dùng Stack)**

*.data*

*array: .word 2, 3, 4, 5, 6*

*N: .word 5*

*newspace: .asciiz ", "*

*.text*

*.globl main*

*main:*

***# Khoi tao ban dau***

*la $t0, array # Load array address*

*li $t1, 0 # i=0*

*lw $t2, N # size of array*

***#Step1:Loop to read items from array to stack***

*pushLoop:*

*lw $t4, ($t0) # get array[i]*

*subu $sp, $sp, 4 # $sp: stack pointer register*

*sw $t4, ($sp) # push to stack*

*add $t1, $t1, 1 # i=i+1*

*add $t0, $t0, 4 # update array address*

*blt $t1, $t2, pushLoop*

***# Khoi tao ban dau***

*la $t0, array # Load array address*

*li $t1, 0 # i=0*

*lw $t2, N # size of array*

***#Step2:Loop to pop items from stack to array***

*popLoop:*

*lw $t4, ($sp) # Get value from stack to t4*

*addu $sp, $sp, 4 # Tang stack*

*sw $t4, ($t0) # Store to array*

*add $t1, $t1, 1 # i=i+1*

*add $t0, $t0,4 # Update array address*

*blt $t1, $t2, popLoop*

***#Step3: print reverse array***

*la $t0, array # Load array address*

*li $t1, 0 # i=0*

*lw $t2, N # size of array*

*printLoop:*

*li $v0, 1*

*lw $a0, ($t0)*

*syscall*

*#print ", "*

*li $v0,4*

*la $a0, newspace*

*syscall*

*addi $t1, $t1, 1 # i=i+1*

*addi $t0, $t0, 4 # tang dia chi array*

*bne $t1, $t2, printLoop*

***#exit:***

*li $v0, 10*

*syscall*

*.end main*

**Bài tập tự thực hành:**

1/ Viết chương trình nhập vào 1 mảng gồm N số nguyên bất kỳ (N nhập từ bàn phím), xuất mảng theo thứ tự ngược lại.

2/ Viết chương trình nhập vào 2 số nguyên a, b. Xuất ra màn hình kết quả của biểu thức: **ab - 12b + 7a** (Dùng stack)

*Gợi ý: -* Bước 1: tính a\*b: push KQ vào stack

* Bước 2: tính -12\*b: push KQ vào stack
* Bước 3: tính 7\*a: push KQ vào stack
* Sau đó POP từng giá trị trong Stack và cộng dồn vào ta sẽ được kết Quả của biểu thức

3/ Viết chương trình nhập vào một chuỗi ký tự, xuất ra chuỗi ngược lại, VD: ABCDEF, xuất ra: FEDCBA

4/ Viết chương trình nhập vào một số nguyên N (VD: N=123). Xuất ra màn hình số ngược lại (VD: 321).