TRƯỜNG ĐH KHOA HỌC TỰ NHIÊN TP.HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN 2

THƯ VIỆN TIME

Thực hiện:

1. Nguyễn Thành Sửu 1712736
2. Nguyễn Tấn Tài 1712742
3. Hoàng Quốc Thịnh 1712790
4. Mai Huy Thông 1712800
5. Lê Trung Tiến 1712811

GVHD: Thầy Lê Viết Long

Tp.Hồ Chí Minh, tháng 4/2019

Mục lục

Cách thức cài đặt một số hàm quan trọng:

- Đầu mỗi hàm luôn có quá trình khai báo stack, sao lưu lại các thanh ghi được sử dụng trong hàm và cuối hàm luôn có quá trình khôi phục lại các thanh ghi đó, xóa stack.

STT	Tên hàm	Tham số truyền vào	Tham số trả về	Cách cài đặt
1	ChieuDaiChuoi (đo chiều dài của một chuỗi ký tự cho trước).	$a0: chứa địa chỉ của một chuỗi ký tự.	$v0: 1 số nguyên cho biết độ dài của chuỗi ký tự đó.	Cho một vòng lặp chạy từ phần tử đầu tiên của chuỗi và so sánh nó với $0, mỗi lần lặp tăng v0 lên 1 đơn vị, khi phần tử của chuỗi bằng 0 thì thoát vòng lặp.
2	DemSoCS (đếm số chữ số của một số nguyên bất kỳ).	$a0: chứa số nguyên cần đếm số chữ số.	$v0: chứa kết quả là 1 số nguyên, đó là số chữ số của số được truyền vào.	Khởi tạo thanh ghi kết quả $v0 có giá trị là 1, liên tục chia $a0 (tham số truyền vào) cho 10 cho đến khi kết quả của phép chia là 0 thì kết thúc. Cặp lệnh div $s0 $t0; mflo $s0 tương ứng với việc ta đang lấy s0 = s0 / 10.
3	itoa (chuyển một số nguyên bất kỳ từ dạng số sang dạng chuỗi).	$a0: số nguyên cần đổi sang dạng ASCII.	$v0: chứa chuỗi sau khi chuyển.	Đầu tiên đếm số chữ số của số này bằng cách gọi hàm DemSoCS và lưu kết quả vào $s0. Tiếp theo, chạy một vòng lặp để tính $t0 = 10 ^ ($s0 - 1) tức $t0 = 10^(số_chữ_số - 1). Tiếp theo, ta cấp phát cho $v0 một vùng nhớ nhất định để lưu chuỗi trả về, sau đó move qua $v1 vì $v0 còn phải dùng nhiều cho việc gọi syscall. Sau khi cấp phát xong vùng nhớ thì chạy một vòng lặp, trong vòng lặp đó ta thực hiện những công việc sau: lấy $a0 = $a0 / $t0 và $s1 = $a0 % $t0, lấy $s1 += 48 để chuyển sang dạng ASCII, đồng thời lưu nó vào trong $v1, tăng địa chỉ $v1 lên 1 đơn vị, cho $t0 /= 10 và lặp lại đến khi $t0 = 0, trong vòng lặp này cũng có một biến

				đếm đặc biệt là biến $t3, nó luôn được gia tăng 1 đơn vị sau mỗi lần lặp, biến này nhằm đếm số địa chỉ mà $v1 đã được cộng thêm, mục đích để khôi phục lại địa chỉ của $v1 sau vòng lặp này (nhắc lại là mỗi lần gán $s1 cho $v1 thì $v1 lại được tăng địa chỉ lên 1 đơn vị). Sau khi chạy xong vòng lặp thì gán lại $v1 về cho $v0 để trả về $v0 (cho phù hợp quy ước), dĩ nhiên là ta không quên trừ đi $t3 để khôi phục lại địa chỉ của $v1 như lúc chưa chạy vòng lặp, để $v1 có thể được in ra bắt đầu từ ký tự đầu tiên của nó
4	atoi (Chuyển một chuỗi biểu diễn số nguyên thành một số nguyên tương ứng).	$a0: chứa địa chỉ của chuỗi cần chuyển đổi.	$v0: một số nguyên chứa giá trị nguyên tương ứng của chuỗi được truyền vào.	Đầu tiên tìm chiều dài của chuỗi cần chuyển đổi bằng hàm ChieuDaiChuoi. Tiếp theo tính $t0 = 10^(Chiều dài chuỗi -1) bằng cách chạy một vòng lặp (chiều dài chuỗi - 1) lần, mỗi lần lặp lấy $t0 *= 10 ($t0 khởi gán là 1). Sau khi tính xong $t0, ta chạy một vòng lặp để tính toán kết quả, vòng lặp đó sẽ làm những công việc sau: load từng ký tự trong $a0 (chuỗi) vào $s1, sau đó lấy $s1 -= 48 để chuyển thành dạng số nguyên, ngay sau đó lấy $s1 *= $t0 và cộng dồn vào thanh ghi kết quả $v1 (được khởi gán = 0); kế đến là tăng $a0 lên 1 địa chỉ, $t0 /= 10, gia tăng $t1 (đóng vai trò index của vòng lặp) lên 1 đơn vị; vòng lặp kết thúc khi $t1 = $s0 (hay index = độ dài chuỗi). Sau khi chạy xong vòng lặp thì chuyển $v1 sang $v0 để trả về $v0 (cho hợp quy ước).
	suu_nhap (nhập dữ liệu từ bàn phím)	Không có	$v0 là mảng ngày, tháng, năm (0($v0), 4($v0), 8($v0)	Cho phép người dùng nhập vào lần lượt ngày, tháng, năm và kiểm tra ngày, tháng, năm nhập vào có hợp lệ hay không bằng cách gọi hàm suu_kiemtrahople. Nếu hợp lệ sẽ trả về kết quả, ngược lại sẽ báo lỗi dữ liệu không hợp lệ.
5	suu_Nhap_ThamSo (nhập dữ liệu có truyền vào tham số)	$a0, $a1, $a2 lần lượt là ngày,	$v0 là mảng ngày, tháng, năm	Tương tự như hàm suu_nhap nhưng có tham số truyền vào chứ không nhập từ bàn phím.

		tháng, năm	(0($v0), 4($v0), 8($v0)	
6	suu_kiemtrahople (kiểm trả dữ liệu ngày, tháng, năm có hợp lệ hay không)	Lấy tham số từ hàm suu_nhap/ suu_Nhap_ThamSo	$v0 (1- hợp lệ, 0 – không hợp lệ)	Kiểm tra lần lượt dữ liệu ngày, tháng, năm có hợp lệ hay không và trả về kết quả vào thanh ghi $v0.
7	LaNamNhuan (kiểm tra xem năm được truyền vào có phải năm nhuận hay không).	$a0: năm truyền vào.	$v0: kết quả (1/0) tương ứng với năm nhuận và không nhuận.	Chạy 2 câu lệnh điều kiện lồng nhau, lấy $t2 = $a0 / 4 và kiểm tra nếu $t2 = 0 thì tiếp tục kiểm tra, ngược lại thì không cần kiểm tra nữa mà returnFalse luôn. Nếu tiếp tục kiểm tra thì tiếp tục lấy $t2 = $a0 % 100, nếu $t2 != 0 thì returnTrue, nếu không thì returnFalse. ReturnTrue đơn thuần chỉ là khởi gán thanh ghi $v0 bằng 1, tương tự với ReturnFalse.
8	hqthinh_cau1 (Thực hiện câu 1).	$a0, với $a0 chứa mảng day month year.	Không có.	Chứa hàm hqthinh_ddmmyyyy để xuất chuỗi ***dd/mm/yyyy*** ra màn hình.
9	hqthinh_ddmmyyyy (in định dạng ***dd/mm/yyyy*** ra màn hình).	Lấy từ hqthinh_cau1.	Không có.	Kiểm tra ngày có phải số có 1 chữ số không (<10), nếu có thì chuyển đến hàm day_1_ditgit, nếu không thực hiện lần lượt các hàm để in ra lần lượt: ngày, dấu gạch chéo, tháng(thực hiện kiểm tra như ngày, nếu là số có 1 chữ số thì chuyển đến hàm month_1_ditgit, nếu không thực hiện in tháng), dấu gạch chéo và năm.
10	day_1_ditgit (in định dạng ***dd/mm/yyyy*** ra màn hình, thực hiện khi ngày < 10).	Lấy từ hqthinh_cau1.	Không có.	Thực hiện in số 0 trước sau đó thực hiện giống như hàm hqthinh_ddmmyyyy bắt đầu từ việc in ngày.
	month_1_ditgit (in định dạng ***dd/mm/yyyy***	Lấy từ hqthinh_cau1.		Thực hiện in số 0 trước sau đó thực hiện giống như hàm hqthinh_ddmmyyyy bắt đầu từ việc in tháng.

	ra màn hình thực hiện khi tháng < 10).			
11	hqthinh_xuat (Thực hiện câu 2)	$a0: chứa mảng ngày, tháng, năm	Không có	Đầu tiên xuất ra menu để lựa chọn định dạng để in ra màn hình. - Nếu chọn A: Đầu tiên in thông báo kết quả ra màn hình sau đó thực hiện như hqthinh_ddmmyyyy nhưng thực hiện in tháng trước, in ngày sau. - Nếu chọn B: Thực hiện in định dạng ***dd Month, year***, cách làm tương tự như các hàm trên trên nhưng in tháng với định dạng ***month*** - Nếu chọn C: Thực hiện in định dạng ***Month dd, year***, cách làm tương tự như B nhưng in tháng trước in ngày sau. - Nếu chọn D: Kết thúc hàm hqthinh_xuat.
12	hqthinh_filecau 1 (in kết quả câu 1 là chuỗi dd/mm/yyyy ra file).	$a0: con trỏ file	Không có	Thực hiện theo thứ tự như hqthinh_ddmmyyyy nhưng gọi thêm hàm itoa để chuyển số dạng int sang dạng string để in ra file.
13	hqthinh_filecau 2a, hqthinh_filecau 2b, hqthinh_filecau 2c, (In kết quả các câu 2a, 2b, 2c)	$a0: con trỏ file	Không có	Thực hiện theo thứ tự như hqthinh_filecau1 nhưng in ra file các định dạng khác từ câu 2A, 2B, 2C.
14	Thong_SoNgay Tu111 (cho biết ngày nhập vào là ngày thứ mấy kể từ ngày 1/1/1).	$a0: mảng ngày, tháng, năm cần biết là ngày thứ mấy	$v0: kết quả là ngày thứ mấy từ ngày 1/1/1	Sử dụng biểu thức: 365 * year + year / 4 – year / 100 + year / 400 + (153 * month – 457) / 5 + day – 306 – 1 để tính. Điều kiện là nếu tháng < 3 thì ta phải giảm năm xuống 1 đơn và tăng tháng lên 12 đơn vị. Đầu tiên ta cần lấy ngày, tháng, năm lưu vào 3 thanh ghi tương ứng là: $s1, $s2, $s3. Tiếp theo kiểm tra $s2 có < 3 hay không, nếu có thì nhảy đến Thong_TangThang_GiamNam

				để điều chỉnh tháng năm cho thỏa điều kiện với biểu thức rồi nhảy đến Thong_TiepTucNeuKhongTang để tiếp túc, ngược lại nhảy đến luôn. Khởi tạo $s0 là biến kết quả cuối cùng, ban đầu gán là 0, $t0 là thanh ghi để gán các giá trị số nguyên để tính toán. Biểu thức được chia nhỏ thành nhiều phần để dễ dàng thực hiện: 365 * year; year / 4; year / 100; year / 400; (153 * month – 457) / 5; + day; – 306; – 1. Lần lượt tính kết quả của từng phần và thực hiện phép tính +/- tương ứng với biểu thức để có kết quả cuối cùng lưu vào $t0. Cuối cùng chuyển dữ liệu từ $t0 sang $v0 và trả về kết quả.
15	Thong_Weekday (cho biết ngày được truyền vào là ngày thứ mấy trong tuần)	$a0: mảng ngày, tháng, năm cần biết là ngày thứ mấy	$v0: kết quả là ngày thứ mấy trong tuần {Mon; Tues, Wed, Thurs, Fri, Sat, Sun}	Đầu tiên ta cần lấy ngày, tháng, năm lưu vào 3 thanh ghi tương ứng là: $s1, $s2, $s3. Sau đó truyền lần lượt $s1, $s2, $s3 vào $a0, $a1, $a2 để gọi hàm Thong_SoNgayTu111 tính số ngày từ ngày 1/1/1 đến ngày đó. Sau khi có kết quả ta chuyển về $t0 và cộng thêm 1 vào $t0 để tiếp tục tính toán. Tại đây, ta lấy kết quả $t0 % 7 và lần lượt đi so sanh với 0 – Chủ nhật; 1 – Thứ hai; 2 – Thứ ba; 3 – Thứ tư; 4 – Thứ năm; 5 – Thứ sáu; 6 – Thứ bảy. Nếu như so sánh kết quả bằng với số mấy thì ta nhảy đến vị trí tương ứng và gán chuỗi kết quả là ngày thứ mấy trong chuỗi đó vào $v0 rồi nhảy đến kết thúc hàm để tải lại dữ liệu và trả về.
16	Thong_GetTime (cho biết khoảng cách giữa 2 chuỗi ngày được truyền vào)	$a0, $a1: mảng 2 ngày cần xem khoảng cách	$v0: kết quả là khoảng cách giữa 2 mảng ngày được truyền vào.	Đầu tiên lấy dữ liệu ngày, tháng, năm từ $a0 lưu lần lượt vào $s1, $s2, $s3, đặc biệt cần lưu lại giá trị ngày, tháng, năm của ngày thứ nhất vào thanh ghi $t2, $t3, $s0 để khôi phục lại sau khi tính ngày thứ hai. Sau đó gọi hàm Thong_SoNgayTu111 để tính là ngày thứ mấy kể từ ngày 1/1/1. Sau đó chuyển kết quả từ $v0 sang $t0 để tính toán sau. Tương tự với dữ liệu từ $a1 và lưu kết quả vào $t1. Khôi phục lại ngày thứ nhất trong $a0. Sau khi tính được khoảng cách của 2 mảng ngày từ $a0 và $a1 được lưu vào $t0, $t1, lấy kết

				quả $t0 - $t1 và lưu vào $t2. So sánh $t2 nếu < 0 thì ta nhảy đến Thong_NhoHon_0 để nhân kết quả với -1 và đi đến trả về; ngược lại thì đi đến trả về. Tại trả về ta chuyển kết quả về $v0 và thực hiện tải lại dữ liệu rồi trả về.
17	NhapXuatFile (Nhập/ xuất File)	$a0 và $a1 tương ứng với 2 chuỗi tên file input và tên file output	Không có giá trị trả về. Kết quả là thay đổi file output	Đầu tiên, mở file để đọc bằng syscall với $v0 = 13 và các tham số $a0, $a1, $a2 tương ứng với tên file, 0, 0 và sau khi mở file thì phải lưu ngay thanh ghi lưu địa chỉ file (tại $v0) vào một thanh ghi $s nào đó (vì $v0 còn phải dùng để gọi nhiều hàm syscall phía sau). Mở file để đọc xong thì tiến hành đọc file bằng syscall với $v0 = 14. Trước tiên thì đọc ngày như sau: đọc chuỗi với độ dài bằng 3 (đọc luôn cả dấu cách) vào trong $a1, lưu sang $s2 (lúc này $s2=day [ascii]), sau đó gọi hàm atoi để chuyển $s2 = day [ascii] thành $t2 = day [int]. Đọc tháng và năm tương tự như vậy, cuối cùng ta sẽ có $t3 = month [int] tương ứng với $s3 = month [ascii] và $t4 = year [int] tương ứng với $s4 = month [ascii]. Với $t2, $t3 và $t4 vừa đọc, ta gọi hàm suu_Nhap_ThamSo để được trả về $v0 với $v0 là 1 mảng 3 phần tử .word $v0, 4($v0) và 8($v0) tương ứng với day, month, year. Ta tiếp tục với việc xuất kết quả của câu 1 và câu 2 ra file, ta sẽ sử dụng hàm XuatFile, hàm nhận vào địa chỉ của file đã mở phía trên và không trả về gì cả, chỉ viết lên file output 3 dạng mà câu 2 yêu cầu. Sau khi hoàn thành xong câu 1 và câu 2 thì ta tiếp tục in câu 3, gọi hàm LaNamNhuan và chạy lệnh rẽ nhánh, nếu là năm nhuận thì in ra thông báo tien_tbNhuan và ngược lại thì in ra thông báo tien_tbKhongNhuan. Tiếp theo ta in ra kết quả câu 4 lên file, chỉ cần gọi hàm Thong_Weekday, move kết quả (là một chuỗi) vào $a0, gọi syscall với $v0 bằng 15. Tiếp theo ta in kết quả câu 5 ra file, cách in như sau, gọi hàm Thong_SoNgayTu111, hàm sẽ trả về kết quả là một số nguyên biểu

				thị là ngày thứ mấy kể từ từ ngày 1/1/1 trong $v0, ta chỉ cần chuyển kết quả này sang dạng chuỗi (sử dụng hàm itoa) và in chuỗi đó ra file. Câu 6 in ra theo cách tương tự như câu 4, tuy nhiên thay vì gọi hàm Thong_Weekday thì ta gọi hàm Tai_CanChi. Câu 7 in tương tự như câu 5, thay vì gọi hàm Thong_SoNgayTu111 thì ta gọi hàm Thong_GetTime. Cuối cùng ta in câu 8, để in kết quả câu này thì ta cũng cần gọi hàm Tai_HaiNamNhuanGanNhat, hàm này trả về $v0 và $v1 tương ứng với 2 năm nhuận gần nhất với năm được truyền vào, vì vậy ta cũng phải chuyển 2 kết quả này sang dạng chuỗi bằng hàm itoa và sau đó in 2 chuỗi này ra file.
18	Tai_CanChi (tìm can chi của năm được truyền vào).	$a0: năm cần biết can chi.	$v0: can. $v1: chi.	Đầu tiên tìm số dư của phép chia năm hiện tại cho 10 và 12 lần lượt lưu vào $t1 và $t2 rồi đi vào Can. Trong Can: căn cứ theo giá trị trong $t1 mà ta tìm được can tương ứng: 0 – Canh, 1- Tân, … 9 – Kỷ. Sau khi tìm được can thì nhảy vào Chi: Trong chi: căn cứ theo giá trị của $t2 mà ta tìm được chi tương ứng: 0 – Thân, 1 – Dậu, … 11 – Mùi. Sau khi tìm được chi thì nhảy vào Tai_Ketthuc. Tại đây, lưu can vào $v0, chi vào $v1 và thực hiện các lệnh cuối thủ tục và trả về nơi gọi hàm.
19	Tai_HaiNamNhuanGanNhat (tìm hai năm nhuận gần nhất với năm được truyền vào).	$a0: truyền vào năm cần biết hai năm nhuận gần nhất.	$v0: năm nhuận nhỏ hơn. $v1: năm nhuận lớn hơn.	Đầu tiên kiểm tra năm được truyền vào, nếu < 4 thì nhảy đến LessThanFour, ngược lại nhảy đến NotLessThanFour. • LessThanFour: hai năm cần tìm là 4 và 8, nhảy đến TwoLeap_return. • NotLessThanFour: kiểm tra năm đó có phải là năm 4: - Nếu đúng, nhảy đến isFour, tại đây hai năm cần tìm là 8 và 12, nhảy đến TwoLeap_return. - Nếu không, nhảy đến BiggerThanFour

				+ BiggerThanFour: Nếu năm đang xét là năm nhuận, nhảy đến pre_Leap để chuẩn bị cho việc tìm đáp số: ✓ pre_Leap: $t0 = Năm – 4 là năm nhuận gần nhất nhỏ hơn và $t1 = Năm + 4 là năm nhuận gần nhất lơn hơn. Sau đó nhảy đến Leap_Smaller để tìm năm nhuận gần nhất nhỏ hơn. ✓ Leap_Smaller: kiểm tra nó có phải là năm nhuận không. Nếu sai thì nhảy đến re_Leap_Smaller để bớt đi 4 và quay lại Leap_Smaller. Nếu đúng thì nhảy đến Leap_Bigger. ✓ Leap_Bigger: kiểm tra nó có phải là năm nhuận không. Nếu sai thì nhảy đến re_Leap_Bigger để tăng thêm 4 và quay lại Leap_Smaller. Nếu đúng thì nhảy đến TwoLeap_return. + Nếu năm đang xét không phải năm nhuận thì nhảy đến pre_NotLeap. Tại đây: $t0 = Năm – 1 là năm nhuận gần nhất nhỏ hơn và nhảy đến NotLeap_Smaller. Tại NotLeap_Smaller: kiểm tra nó có phải là năm nhuận không. Nếu sai thì nhảy đến re_NotLeap_Smaller để bớt đi 1 và quay lại Leap_Smaller. Nếu đúng thì: $t1 = $t0 + 4 và nhảy đến NotLeap_Bigger. Tại NotLeap_Bigger: kiểm tra nó có phải là năm nhuận không. Nếu sai thì nhảy đến re_NotLeap_Bigger để tăng thêm 4 và quay lại Leap_Smaller. Nếu đúng thì nhảy đến TwoLeap_return. + TwoLeap_return: gán $v0 = $t0 là năm nhuận nhỏ hơn tìm được và $v1 = $t1 là năm nhuận lớn hơn tìm được. Sau đó thực hiện các lệnh cuối thủ tục và trả về nơi gọi hàm.

Quy tắc viết và gọi hàm trong MIPS:

- Các thanh ghi được MIPS hỗ trợ để lưu trữ dữ liệu trong thủ tục:
  - Đối số input (argument input): $a0, $a1, $a2, $a3
  - Kết quả trả về (return …): $v0, $v1
  - Biến cục bộ trong thủ tục: $s0, $s1, … $s7
  - Địa chỉ quay về (return address) $ra
  - Nếu có nhu cầu lưu nhiều dữ liệu (đối số, kết quả trả về, biến cục bộ) hơn số lượng thanh ghi kể trên thì sử dụng ngăn xếp (stack)..
- Trước khi gọi thủ tục: Cần truyền các đối số vào các thanh ghi đối số ($a0, $a1, $a2, $a3).
- Nguyên tắc khi thực thi thủ tục:
  - Nhảy đến thủ tục bằng lệnh jal và quay về nơi trước đó đã gọi nó bằng lệnh jr $ra
  - 4 thanh ghi chứa đối số của thủ tục: $a0, $a1, $a2, $a3
  - Kết quả trả về của thủ tục chứa trong thanh ghi $v0 (và $v1 nếu cần)
  - Nguyên tắc sử dụng các thanh ghi:
    - Nếu thủ tục R gọi thủ tục E thì:

▪ R phải lưu vào stack các thanh ghi tạm có thể bị sử dụng trong E trước khi gọi lệnh jal E (như lệnh goto E trong C/C++).
▪ E phải lưu lại giá trị các thanh ghi lưu trữ nếu nó muốn sử dụng các thanh ghi này và trước khi kết thúc E sẽ khôi phục lại giá trị của chúng.
▪ Thủ tục gọi R (caller) và Thủ tục được gọi E (callee) chỉ cần lưu các thanh ghi tạm / thanh ghi lưu trữ mà nó muốn dùng, không phải tất cả các thanh ghi!
    - Ví dụ: Trong bài đồ án có hàm kiểm tra năm nhuận: LaNamNhuan:
▪ Trước khi hàm được gọi ở nơi nào đó cần truyền đối số là năm cần kiểm tra vào thanh ghi $a0.
▪ Ở đầu hàm cần khai báo và sao lưu các thanh ghi trong hàm cần sử dụng ($a0, $t1, $t2) và $ra vào stack.
▪ Kết quả trả về được lưu vào thanh ghi $v0 (là true – 1 hay false – 0).
▪ Trước khi kết thúc hàm cần khôi phục lại các thanh ghi đã sao lưu ($a0, $t1, $t2 và $ra) và khôi phục stack.
▪ Dùng lệnh ***jr $ra*** để trở về địa chỉ nơi gọi hàm.

Tài liệu tham khảo:

[1] othanh.blogspot.com/2016/04/thuat-toan-tinh-khoang-cach-giua-2-ngay.html