Cách cài đặt hàm ChieuDaiChuoi (Đo chiều dài của một chuỗi ký tự cho trước): Hàm nhận vào a0, với a0 chứa địa chỉ của một chuỗi ký tự, trả về v0 với v0 là 1 số nguyên cho biết độ dài của chuỗi ký tự đó. Cách cài đặt: 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.

Cách cài đặt hàm DemSoCS (đếm số chữ số của một số nguyên bất kỳ): Hàm truyền vào a0, với a0 chứa số nguyên cần đếm số chữ số; v0 với v0 chứa kết quả là 1 số nguyên, đó là số chữ số của số được truyền vào). Cách cài đặt: 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.

Cách cài đặt hàm itoa (chuyển một số nguyên bất kỳ từ dạng số sang dạng chuỗi): Hàm truyền vào a0, với a0 là số nguyên cần đổi sang dạng ASCII, và trả về v0 với v0 chứa chuỗi sau khi chuyển. Cách cài đặt: 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 nhắm 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.

Cách cài đặt hàm atoi (Chuyển một chuỗi biểu diễn số nguyên thành một số nguyên tương ứng): Hàm nhận vào a0, với a0 chứa địa chỉ của chuỗi cần chuyển đổi, trả về v0 với v0 là một số nguyên chứa giá trị nguyên tương ứng của chuỗi được truyền vào. Cách cài đặt như sau: 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).

Cách cài đặt hàm LaNamNhuan (kiểm tra xem năm được truyền vào có phải năm nhuận hay không): Hàm nhận vào tham số a0 với a0 là năm truyền vào, trả về v0 với v0 là kết quả (1/0) tương ứng với năm nhuận và không nhuận. Cách cài đặt như sau: 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.

Cách cài đặt hàm NhapXuatFile: Hàm truyền vào a0 và a1 tương ứng với 2 chuỗi tên file input và tên file output; không có giá trị trả về, chỉ có file output được thay đổi. Cách cài đặt như sau: 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 NhapThamSo (sẽ được mô tả ở phần khác trong báo cáo) 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 (sẽ được mô tả riêng), 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. Tiêp theo ta in kết quả câu 5 ra file, cách in như sau, gọi hàm SoNgayTu111 (sẽ được mô tả sau), hàm sẽ trả về kết quả là một số nguyên biểu thị số ngày 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 Can\_Chi. Câu 7 in tương tự như câu 5, thay vì gọi hàm SoNgayTu111 thì ta gọi hàm KhoangCachNgay. Cuối cùng ta in câu 8, để in kết quả câu này thì ta cũng cần gọi hàm NamNhuanGanNhat, 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.