TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM

KHOA CÔNG NGHỆ THÔNG TIN

----📖----

BÁO CÁO ĐỒ ÁN 2

THƯ VIỆN TIME

Nhóm sinh viên thực hiện:

Vũ Thành Đạt	**1612087**
Đặng Thái Gia Thuận	**1712173**
Lê Đức Thuận	**1712805**
Huỳnh Thanh Khải Trân	**1712828**
Nguyễn Phượng Vỹ (*)	**1712929**

(*): Nhóm trưởng.

TP.Hồ Chí Minh, ngày 23 tháng 04 năm 2019

I. Nội dung.

1. Đánh giá mức độ hoàn thành.

STT	Nội dung		Tỷ lệ hoàn thành (%)	Ghi chú
1	Xuất chuỗi TIME theo định dạng DD/MM/YYYY		100	
2	Chuyển đổi chuỗi TIME thành một trong các định dạng	MM/DD/YYYY	100	
		Month DD, YYYY	100	
		DD Month, YYYY	100	
3	Kiểm tra năm trong chuỗi TIME có phải là năm nhuận không		100	
4	Cho biết ngày vừa nhập là ngày thứ mấy trong tuần		100	
5	Cho biết ngày vừa nhập là ngày thứ mấy kể từ ngày 01/01/0001		100	
6	Cho biết can chi của năm vừa nhập		100	
7	Cho biết khoảng thời gian giữa chuỗi TIME_1 và TIME_2		100	
8	Cho biết 2 năm nhuận gần nhất với năm trong chuỗi time		100	
9	Nhập input từ file input.txt xuất kết quả toàn bộ các chức năng trên ra file output.txt		100	Nếu không có File input.txt thì sẽ xuất ra file output.txt nếu có truyền vào giá trị Time2 (gọi yêu cầu 7 trước khi gọi yêu cầu 9)
10	Kiểm tra bộ dữ liệu đầu vào khi nhập		100	

Bảng 1. Bảng đánh giá chi tiết các yêu cầu của đồ án 2

2. Mô tả các hàm chính.

Quy định chung: Lưu các biến (thay đổi trong hàm) trước khi bước vào giai đoạn xử lý trong thân hàm để có thể tận dụng hàm dễ dàng hơn và quản lí biến dễ hơn, trả lại giá trị lưu trước khi kết thúc hàm.

2.1. Hàm int Day(char* TIME).

- Mô tả: hàm lấy ngày dạng số nguyên từ chuỗi "DD/MM/YYYY".

- Input: chuỗi TIME có kiểu char* với định dạng "DD/MM/YYYY".

- Output: Trả về ngày kiểu số nguyên int.

- Thuật toán, chi tiết xử lý các bước:

• Cấp phát vùng nhớ (nhãn 'day').

• Lưu các giá trị của các thanh ghi có thể thay đổi vào vùng nhớ stack.

• Dùng thanh ghi tạm $t0 lưu chuỗi $a0 đầu vào, $v0 lưu địa chỉ của nhãn 'day', $v0 lúc này là chuỗi rỗng.

• Lấy 2 kí tự đầu của $t0, tức là "DD" của $t0 ("DD/MM/YYYY") lưu vào $v0.

• Gọi hàm atoi (tham số $a0 gán bằng $v0: "DD") để chuyển chuỗi "DD" thành số nguyên, sau khi gọi atoi, $v0 lưu giá trị ngày có kiểu số nguyên (int).

• Trả lại các giá trị đã lưu trong vùng nhớ stack cho các thanh ghi trước đó và kết thúc hàm.

2.2. Hàm int Month(char* TIME).

- Mô tả: hàm lấy tháng dạng số nguyên từ chuỗi "DD/MM/YYYY".

- Input: chuỗi TIME có kiểu char* với định dạng "DD/MM/YYYY".

- Output: Trả về tháng kiểu số nguyên int.

- Thuật toán, chi tiết xử lý các bước:

• Cấp phát vùng nhớ (nhãn 'month').

• Lưu các giá trị của các thanh ghi có thể thay đổi vào vùng nhớ stack.

• Dùng thanh ghi tạm $t0 lưu chuỗi $a0 đầu vào, $v0 lưu địa chỉ của nhãn 'month', $v0 lúc này là chuỗi rỗng.

• Lấy 2 kí tự thứ 3 và 4 của $t0, tức là “MM” của $t0 (“DD/MM/YYYY”) lưu vào $v0.

• Gọi hàm atoi (tham số $a0 gán bằng $v0: “MM”) để chuyển chuỗi “MM” thành số nguyên, sau khi gọi atoi, $v0 lưu giá trị tháng có kiểu số nguyên (int).

• Trả lại các giá trị đã lưu trong vùng nhớ stack cho các thanh ghi trước đó và kết thúc hàm.

2.3. Hàm int Year(char* TIME).

- Mô tả: hàm lấy năm dạng số nguyên từ chuỗi “DD/MM/YYYY”.

- Input: chuỗi TIME có kiểu char* với định dạng “DD/MM/YYYY”.

- Output: Trả về năm kiểu số nguyên int. - Thuật toán, chi tiết xử lý các bước:

• Cấp phát vùng nhớ (nhãn ‘year’).

• Lưu các giá trị của các thanh ghi có thể thay đổi vào vùng nhớ stack.

• Dùng thanh ghi tạm $t0 lưu chuỗi $a0 đầu vào, $a0 lưu địa chỉ của nhãn ‘year’, $a0 lúc này là chuỗi rỗng, $t1 gán bằng $a0 để lưu địa chỉ nhãn ‘year’ (đầu chuỗi ‘year’).

• Chạy vòng lặp để duyệt hết các kí tự “YYYY” của $t0 (“DD/MM/YYYY”) lưu vào $a0, mỗi lần lặp, giá trị $a0 được tăng lên và cuối cùng lưu địa chỉ cuối chuỗi ‘year’ nên sau vòng lặp phải gán $a0 về lại $t1 địa chỉ đầu chuỗi ‘year’. Gọi hàm atoi (tham số $a0: “YYYY”) để chuyển chuỗi “YYYY” thành số nguyên, sau khi gọi atoi, $v0 lưu giá trị năm có kiểu số nguyên (int).

• Trả lại các giá trị đã lưu trong vùng nhớ stack cho các thanh ghi trước đó và kết thúc hàm.

2.4. Hàm int atoi(char* num).

- Mô tả: hàm chuyển số dạng chuỗi thành số nguyên.

- Input: số dạng chuỗi cần chuyển kiểu char*.

- Output: giá trị số nguyên của chuỗi sau khi chuyển kiểu int.

- Thuật toán, chi tiết xử lý các bước:

• Lưu các giá trị của các thanh ghi có thể thay đổi vào vùng nhớ stack.

• $v0 là thanh ghi lưu giá trị số nguyên sau khi chuyển, gán $v0 = $0, gán $t0 = $a0 để thao tác trên $t0.

• Duyệt vòng lặp kiểm tra từng kí tự của $t0 nếu không là kí tự số ‘0’ … ‘9’ thì kết thúc, ngược lại nếu là kí tự số ‘0’ … ‘9’ thì trừ 48 để chuyển kí tự đó về số nguyên ($t1).

• Mỗi lần lặp $v0 = $v0 *10 + $t1.

• Trả lại các giá trị đã lưu trong vùng nhớ stack cho các thanh ghi trước đó và kết thúc hàm.

2.5. Hàm input (char* input(char* day, char *month, char* year)).

- Công dụng: Lần lượt nhập ngày, tháng, năm (kiểu chuỗi), kiểm tra hợp lệ trong quá trình nhập, nếu không hợp lệ sẽ yêu cầu nhập lại. - Thuật toán, chi tiết xử lý các bước:

• Lưu các biến tạm sẽ sử dụng vào stack.

• Nhãn inputNgay: Nhập chuỗi ngày.

• Nhãn checkDayInput: Lấy từng kí tự chuỗi ngày vừa nhập, nếu kí tự không thuộc [0,9] thì báo lỗi, gọi nhập lại, nếu chuỗi ngày hợp lệ thì đi đến nhãn outputNgay.

• Nhãn ErrorInNgay: Xuất chuỗi báo lỗi, yêu cầu nhập lại và quay trở lại nhãn inputNgay để nhập lại chuỗi ngày.

• Nhãn outputNgay: Chuyển chuỗi vừa nhập kiểu char* thành dạng int.

• Nhãn inputThang: Nhập chuỗi tháng.

• Nhãn checkMonthInput: Lấy từng kí tự chuỗi tháng vừa nhập, nếu kí tự không thuộc [0,9] thì báo lỗi, gọi nhập lại, nếu chuỗi tháng hợp lệ thì đi đến nhãn outputThang.

• Nhãn ErrorInThang: Xuất chuỗi báo lỗi, yêu cầu nhập lại, quay trở lại nhãn inputThang để nhập lại chuỗi tháng.

• Nhãn outputThang: Chuyển chuỗi vừa nhập kiểu char* sang int.

• Nhãn inputNam: Nhập chuỗi năm.

• Nhãn checkYearInput: Lấy từng kí tự chuỗi năm vừa nhập, nếu kí tự không thuộc [0,9] thì báo lỗi, gọi nhập lại, nếu chuỗi năm hợp lệ thì đi đến nhãn outputNam.

• Nhãn ErrorInNam: Xuất chuỗi báo lỗi, yêu cầu nhập lại, quay trở lại nhãn inputNam để nhập lại chuỗi năm.

• Nhãn outputNam: Chuyển chuỗi vừa nhập kiểu char* sang int. Sau đó kiểm tra ngày, tháng, năm đã nhập có hợp lệ không (nếu không hợp lệ thì quay trở lại nhãn

ErrorInNgay – Nhập lại ngày, tháng, năm từ đầu; nếu hợp lệ chuyển ngày, tháng, năm đã nhập thành chuỗi theo định dạng DD/MM/YYYY hoặc "\0" nếu chuỗi rỗng).

• Trả lại các biến lưu vào stack, trả về.

2.6. Hàm int CheckDate(int day, int month, int year).

- Mô tả: Kiểm tra ngày, tháng năm có hợp lệ hay không.

- Input: day, month, year tương ứng là 3 giá trị ngày, tháng, năm do người dung nhập vào. Với $a0 day, $a1 month, $a2 year.

- Output: Trả về 1 nếu hợp lệ hoặc 0 nếu không hợp lệ.

- Thuật toán, chi tiết xử lý các bước:

• Lưu các biến tạm sẽ sử dụng vào stack.

• Kiểm tra ngày, tháng, năm tương ứng với nhau có đúng không.

• Nhãn end_check: Trả lại các biến tạm lưu vào stack, trả về.

2.7. Hàm int DayOfMonth(int year, int month).

- Mô tả: Tính số ngày của tháng.

- Input: month, year là tháng và năm cần kiểm tra.

- Output: Trả về số ngày của tháng đầu vào.

- Thuật toán, chi tiết xử lý các bước:

• Lưu các biến tạm sẽ sử dụng vào stack.

• Lần lượt so sánh tháng là tháng thứ mấy để trả về số ngày của tháng đó. Đặc biệt, nếu là tháng 2 thì kiểm tra năm nhuận (isLeapYear trả về 0/1, sau đó 0/1+28 sẽ được ngày của tháng 2) để trả về 28/29 ngày.

• Nhãn end_DayOfMonth: Trả lại các biến tạm lưu vào stack, trả về.

2.8. Hàm char* Date(int day, int month, int year, char* TIME).

- Mô tả: Xuất chuỗi TIME theo định dạng mặc định DD/MM/YYYY.

- Input: day, month, year tương ứng là 3 giá trị ngày, tháng, năm do người dung nhập vào, TIME trỏ đến vùng nhớ lưu trữ kết quả chuỗi ngày tháng đã định dạng.

- Output: $v0 là địa chỉ trỏ đến vùng nhớ lưu trữ kết quả chuỗi ngày tháng năm theo định dạng DD\MM \YYYY.

Cách làm: Day và month đều có tối đa 2 chữ số nên ta lấy giá trị day, month chia cho 10. Chữ số đầu tiên là phần nguyên, chữ số thứ 2 là phần dư. Lấy 2 kí tự ngày store byte vào 0($a3) và 1($a3) , tiếp tục ta store byte vào 2($a3) kí tự '/' (47). Cứ như vậy đến tháng. Năm thì ta làm tương tự nhưng chia cho 1000 rồi 100 rồi 10 để lấy đı 4 chữ số và store byte đến 9($a3). Sau đó trả về kết quả là địa chỉ của thanh ghi $a3.

2.9. Hàm char* Convert(char* TIME, char type).

- Mô tả: Chuyển đổi kiểu định dạng của chuỗi TIME.

- Input: TIME trỏ đến vùng nhớ lưu giá trị chuỗi ngày tháng cần định dạng, type là kiểu định dạng muốn chuyển (type = 'A': định dạng MM/DD/YYYY, type = 'B': định dạng Month DD, YYYY; type = 'C': định dạng DD Month, YYYY).

- Output: Trả về chuỗi result theo định dạng yêu cầu. - Thuật toán, chi tiết xử lý các bước:

• Lưu các biến tạm sẽ sử dụng vào stack.

• Nhãn Convert: Kiểm tra type thuộc kiểu gì để đi tới đúng nhãn xử lý định dạng kiểu ấy (nếu type='A' đi đến nhãn C2A, nếu type='B' đi đến nhãn C2B, nếu type='C' đi đến nhãn C2C).

• Nhãn C2A: Lần lượt lấy chuỗi tháng, ngày, năm từ chuỗi đầu vào đưa vào chuỗi result, đồng thời chèn dấu '/' để được chuỗi theo định dạng "MM/DD/YYY".

• Nhãn C2B: Lấy chuỗi tháng, sau đó gọi textMonth để lấy tên tiếng Anh của tháng trong chuỗi đầu vào. Tiếp theo lấy chuỗi ngày, năm, đồng thời chèn các kí tự ',' và ' ' theo thứ tự để được chuỗi theo định dạng "Month DD, YYYY".

• Nhãn C2C: Lấy chuỗi tháng, sau đó gọi textMonth để lấy tên tiếng Anh của tháng trong chuỗi đầu vào, lấy chuỗi ngày và năm đồng thơi chèn các kí tự ' ' và ',' theo thứ tự để được chuỗi thjeo định dạng DD Month, YYYY.

• Nhãn end_Convert: Trả lại các biến tạm lưu vào stack, trả về.

2.10. Hàm char* textMonth(int month).

- Mô tả: Tìm tên tiếng Anh của tháng.

- Input: giá trị kiểu nguyên của tháng.

- Output: Con trỏ trỏ đến vùng nhớ lưu giá trị là chuỗi tên tiếng Anh của tháng.

- Thuật toán, chi tiết xử lý các bước:

• Nhãn textMonth: Lưu các biến tạm sẽ sử dụng vào stack. Nếu tháng truyền vào bằng 1, nhảy tới nhãn January. Lần lượt kiểm tra như thế cho tới tháng 12, với mỗi tháng trị nhảy tới nhãn tương ứng. Ở mỗi nhãn tương ứng với từng tháng, gán kết quả trả về trỏ tới vùng nhớ trỏ đến chuỗi tên tiếng Anh của tháng rồi nhảy tới nhãn End_Month.

• Nhãn: End_Month: Trả các biến đã lưu vào stack, trả về.

2.11. Hàm int isLeapYear (int year).

- Mô tả: Kiểm tra năm nhuận.

- Input: giá trị kiểu nguyên của năm cần kiểm tra.

- Output:

  • 0: nếu giá trị nguyên của năm truyền vào không phải năm nhuận.

  • 1: nếu giá trị nguyên của năm truyền vào là năm nhuận

- Thuật toán, chi tiết xử lý các bước:

• Nhãn isLeapYear: Lưu các biến tạm sẽ sử dụng vào stack. Lấy năm truyền vào chia cho 400, nếu chia hết thì nhảy tới nhãn true, ngược lại, nếu năm truyền vào chia hết cho 100, nhảy tới nhãn false; ngược lại, nếu năm truyền vào không chia hết cho 4, nhảy tới nhãn false.

Thuật toán C++:

```
if (((year % 4 == 0) && (year % 100!= 0)) || (year%400 == 0))
return 1;
else return 0;
```

• Nhãn true: Gán kết quả bằng 1, nhảy đến nhãn end_isLeapYear.

• Nhãn false: Gán kết ủa bằng 0.

• Nhãn: end_isLeapYear: Trả các biến đã lưu vào stack, trả về.

2.12. Hàm int LeapYear(char* TIME).

- Mô tả: Kiểm tra năm nhuận.

- Input: TIME trỏ đến vùng nhớ lưu giá trị ngày tháng năm cần xử lý.

- Output:

  • 0: nếu chuỗi TIME không phải năm nhuận.

• 1: nếu chuỗi TIME của năm truyền vào là năm nhuận

- Thuật toán, chi tiết xử lý các bước:

• Nhãn LeapYear: Lấy chuỗi TIME làm tham số đầu vào cho hàm Year. Gọi hàm Year để lấy giá trị nguyên của năm trong chuỗi TIME, với kết quả trả về là giá trị nguyên của năm sẽ được dùng làm tham số để gọi hàm isLeapYear. Kết quả trả về sau khi gọi hàm cũng chính là giá trị 0 hoặc 1.

• Kết thúc hàm: Trả lại các biến lưu vào stack, trả về.

2.13. Hàm char* WeekDay(char* TIME).

- Mô tả: Cho biết giá trị ngày trong chuỗi TIME là thứ mấy trong tuần.

- Input: TIME là con trỏ trỏ đến vùng nhớ lưu giá trị ngày tháng cần xử lý.

- Output: Trả về con trỏ trỏ đến vùng nhớ lưu chuỗi tên của thứ trong tuần.

- Thuật toán, chi tiết xử lý các bước:

• Nhãn WeekDay: Lưu các biến tạm sẽ sử dụng vào stack. Lưu thanh ghi $ra vào stack vì sẽ có gọi hàm.

• Với chuỗi TIME là giá trị đầu vào để gọi lần lượt các hàm Day, Month, Year. Các kết quả trả về của các hàm vừa gọi sẽ lần lượt là giá trị. nguyên của ngày, tháng, năm và sẽ được lưu vào các biến tạm.

• Nếu tháng >= 3 nhảy tới nhãn LoopWeek. Ngược lại, tháng=tháng+12, năm=năm-1.

• Nhãn LoopWeek:

• Lần lượt thực hiện các phép toán theo công thức: Thứ=(ngày+2*tháng+(3*tháng+3) div 5 + năm + (năm div 4)) mod 7.

• Kết quả thứ sẽ là giá trị nguyên thuộc tập {0,1,2,3,4,5,6} tương ứng sẽ nhảy tới các nhãn isCN, isTh2, isTh3, isTh4, isTh5, isTh6, isTh7.

• Với lần lượt các nhãn trên, thì kết quả trả về là con trỏ trỏ đến vùng nhớ chứa chuỗi tương ứng thuộc tập. {“Chu Nhat”,”Thu hai”, ”Thu ba”, ”Thu tu”, ”Thu nam”, ”Thu sau”, ”Thu bay” } và nhảy tới nhãn endD.

• Nhãn endD: Trả các biến đã lưu vào stack, trả về.

2.14. Hàm int FullDate(char* TIME).

- Mô tả: Cho biết ngày vừa nhập là ngày thứ mấy kể từ ngày 01/01/0001

- Input: TIME là con trỏ trỏ đến vùng nhớ lưu giá trị ngày tháng cần xử lý.

- Output: Trả về con trỏ trỏ đến vùng nhớ lưu khoảng cách từ TIME đến 1/1/1.

- Thuật toán, chi tiết xử lý các bước:

• Nhãn FullDate: Lưu các biến tạm sẽ sử dụng vào stack.

• Với chuỗi TIME là giá trị đầu vào để gọi lần lượt các hàm Day, Month, Year. Các kết quả trả về của các hàm vừa gọi sẽ lần lượt là giá trị. nguyên của ngày, tháng, năm và sẽ được lưu vào các biến tạm.

• Sử dụng công thức tính: khoảng cách = year*365 + year/4 - year/100 + year/400 + (153*month - 457)/5 + day -306

• Trả các biến đã lưu vào stack, trả về.

2.15. Hàm char* Can (char* TIME)

- Mô tả : Từ số năm từ TIME cho biết can của năm đó là gì.

- Input TIME.

- Output : Can tương ứng.

- Thuật toán, chi tiết xử lý các bước:

• Gọi hàm lấy số năm từ TIME.

• Gán biến tạm = 10, số năm cộng thêm 6.

• Lấy số năm chia cho 10

• Kết quả phép chia dư (0, 1, …, 9) tương ứng Can (“Giap”, “At”, …, “Quy”).

• End_Can : trả lại các giá trị đã lưu trong vùng nhớ stack cho các thanh ghi trước đó và kết thúc hàm.

2.16. Hàm char* Chi (char* TIME)

- Mô tả : Từ số năm từ TIME cho biết năm đó là năm gì (Tý, Sửu, Dần, Meo, Thìn, Tỵ, Ngọ, Mao, Thân, Dậu, Tuất, Hợi).

- Input TIME.

- Output : Chi tương ứng.

- Thuật toán, chi tiết xử lý các bước:

• Gọi hàm lấy số năm từ TIME.

• Gán biến tạm = 12, số năm cộng thêm 8.

• Lấy phần dư của số năm chia cho 12.

• Kết quả phép chia dư (0, 1, …, 11) tương ứng Chi (“Ty”, “Suu”, …, “Hoi”).

• End_Chi : trả lại các giá trị đã lưu trong vùng nhớ stack cho các thanh ghi trước đó và kết thúc hàm.

2.17. Hàm int DateDiff (char* TIME1, char* TIME2).

- Mô tả: Tính khoảng thời gian cách biệt (ngày) của chuỗi TIME_1 và TIME_2.

- Input: TIME_1 và TIME_2 là con trỏ trỏ đến vùng nhớ lưu giá trị ngày tháng năm cần xử lý.

- Output: Số ngày cách biệt.

- Thuật toán, chi tiết xử lý các bước:

• Nhãn DateDiff: Lưu các biến tạm sẽ sử dụng vào stack.

• Với TIME1 và TIME2 là đầu vào để gọi hàm FullDate để tính tổng số ngày kể từ ngày 1/1/1 .

• Lấy số ngày đã tính của FullDate(TIME1) – FullDate(TIME2) (nếu TIME1>TIME2) hoặc FullDate(TIME2) – FullDate(TIME1) (nếu TIME1<TIME2)

• Nhãn End_DateDiff: Trả các biến đã lưu vào stack, trả về.

2.18. Hàm void TwoNearestLeapYear(char* TIME).

- Mô tả: In ra 2 năm nhuận gần nhất với năm trong chuỗi TIME.

- Input: TIME là con trỏ trỏ đến vùng nhớ lưu giá trị ngày tháng cần xử lý.

- Output: In ra 2 năm nhuận gần nhất với năm trong chuỗi TIME.

- Thuật toán, chi tiết xử lý các bước:

• Nhãn TwoNearestLeapYear:

• Lưu các biến tạm sẽ sử dụng vào stack.

• Lưu thanh ghi $ra vào stack vì sẽ có gọi hàm.

• Lấy chuỗi TIME là giá trị đầu vào để gọi hàm Year. Kết quả trả về sẽ là giá trị nguyên của năm trong chuỗi TIME.

• Nhãn forLY1: dùng để tìm năm nhuận gần nhất nhỏ hơn năm trong chuỗi TIME

• Năm=năm – 1.

• Kiểm tra xem năm đang xét có phải năm nhuận hay không bằng cách gọi hàm LeapYear với tham số là chuỗi ngày tháng năm vừa nhận được từ hàm Date.

• Nếu là năm nhuận, nhảy tới nhãn forLY2.

• Nhãn forLY2: dùng để tìm năm nhuận gần nhất lớn hơn năm trong chuỗi TIME và lưu kết quả LY1 vào thanh ghi tạm $t0 và trả lại kết quả là giá trị nguyên của năm trong chuỗi TIME.

•Nhãn forLY2Next: dùng để tìm năm nhuận gần nhất lớn hơn năm trong chuỗi TIME.

+ Năm=năm + 1.

+ Kiểm tra xem năm đang xét có phải năm nhuận hay không bằng cách gọi hàm LeapYear với tham số là chuỗi ngày tháng năm vừa nhận được từ hàm Date.

+ Nếu là năm nhuận, nhảy tới nhãn exitTNLY

+ Tiếp tục vòng lặp bằng cách nhảy lại tới nhãn forLY2Next.

• Nhãn exitTNLY: Trả các biến đã lưu vào stack, trả về.

2.19. Hàm int ItoA (int, char*)

- Mô tả: hàm chuyển số nguyên sang chuỗi string

- Input: số nguyên($a0)

- Output: mảng số nguyên dạng ký tự ($a1)

- Thuật toán, chi tiết xử lý các bước:

• Khai báo bộ nhớ $a1, $t1

• Kiểm tra $a0, nếu khác 0 thì gọi ItoA.non_zero

• Hàm itoA.non_zero:

+ Nếu $a0>0, gọi hàm ItoA.recurse

+ Nếu $a0 <0, thì gán ‘-‘ vào đầu mảng $a1, sau đó dùng neg để chuyển $a0 về số đối là số dương

+ Lưu $t1 vào $a1

• Hàm ItoA.recurse:

+ Hàm đệ quy để gọi ghi từng ký tự vào $s1 đến hết chiều dài chuỗi số nguyên

+ Dùng phép dịch bit các bit trong $a0, mfhi để lấy 1 bit đang xét lưu vào $s1, sau đó gọi ItoA.write đề gán $s1 vào $t1

• Hàm ItoA.continue: Lặp hàm ItoA.recurse (chạy tự nhiên theo tứ tự từ trên xuống dưới chứ không gọi hàm)

• Hàm ItoA.write:

+ Gán bit $s1 đang giữ vào byte trống cuối cùng.

+ Byte trống kế tiếp gán cho $zero để chờ lưu tiếp hoặc kết thúc.

• Hàm ItoA.exit: Reset lại stack và chuyển sang thao tác tiếp theo

2.20. Hàm void XuLyFile()

Bước 1: Mở file input.txt ở chế độ read có đường dẫn tương đối là nơi chứ file project2.asm đọc file và ghi vào $a1, sau đó đóng file input.txt

Bước 2: Load file, đọc chuỗi trong $a1 gồm 2 dòng, thay khoảng trắng bằng '/' ta được 2 chuỗi time1 và time2 chuẩn DD/MM/YYYY

Bước 3: Mở file output.txt ở chế đọ write, File descriptor nằm trong thanh ghi $t5

Bước 4: Check Date, kiểm tra xem ngày trong time1 và time2 có hợp lệ hay không bằng hàm CheckDate

- int CheckDate(int day,int month,int year)

- input: $a0 day. $a1 month. $a2 year

- output: 1 if valid date else 0

Nếu time1 sai, time2 sai hoặc cả 2 sai thì in vào file output.txt thông báo và chuyển sang Bước 6.

Bước 5: Ta in kết quả từ YC1 đến YC8 vào output.txt bằng hàm yc9wf

5.1 Ghi time1 vào ouput.txt

5.2

a. gán giá trị $a1 = 65 (ký tự 'A'), gọi hàm Convert và in kết quả vào output.txt

b. gán giá trị $a1 = 66 (ký tự 'B'), gọi hàm Convert và in kết quả vào output.txt

c. gán giá trị $a1 = 67 (ký tự 'C'), gọi hàm Convert và in kết quả vào output.txt

5.3 Gán $a0 là time1 và gọi hàm LeapYear – nếu trả về 1 thì in vào output.txt là năm nhuận, trả về 0 thì in vào output.txt là năm thường

5.4 Gán $a0 là time1 và gọi hàm WeekDay, sau đó in kết quả trả về vào output.txt

5.5 Gán $a0 là time1 và gọi hàm FullDate, chuyển kết quả trả về có dạng int sang char bằng hàm ItoA sau đó in vào output.txt

5.6 Gán $a0 là time1 và gọi hàm Can, Chi, Sau đó in kết quả trả về vào output.txt

5.7 Gán $a0 là time1, $a1 là time2 và gọi hàm DateDiff, chuyển kết quả trả về có dạng int sang char bằng hàm ItoA sau đó in vào output.txt

5.8 Gán $a0 là time1, Gọi hàm TwoNextLeapYear, chuyển kết quả trả về trong $v0 và $v1 từ int sang char bằng hàm ItoA và in vào output.txt

Bước 6: Đóng file và kết thúc xử lý File.

3. Mô tả các yêu cầu trong hàm Menu

3.1 YC1: Xuất chuỗi TIME vừa đọc từ file theo định dạng DD/MM/YYYY.

3.2 YC2: Chuyển đổi chuỗi TIME thành một trong các định dạng sau:

A. MM/DD/YYYY.

B. Month DD, YYYY.

C. DD Month, YYYY.

- Lựa chọn nhập vào (A , B , C) sau đó với lựa chọn đúng sẽ vào nhãn continueYC2.
- Trong nhãn continueYC2 sẽ cho tiếp tục tới nhãn Convert.
- Convert sẽ trả về giá trị adate đúng định dạng yêu cầu.
- In kết quả ra màn hình.

3.3 YC3: Kiểm tra năm trong chuỗi TIME có phải là năm nhuận không.

- Sao chép TIME vào $a0.
- Gọi hàm LeapYear.
- Giá trị LeapYear trả về là 0 thì nhảy vào nhãn NotLeap.
- NotLeap in ra màn hình “Nam khong nhuan\n”.
- Giá trị LeapYear trả về khác 0 thì in ra màn hình “Nam nhuan\n”.

3.4 YC4: Cho biết ngày vừa nhập là ngày thứ mấy trong tuần.

• Sao chép TIME vào $a0.

• Gọi hàm WeekDay.

• Kết quả của hàm WeekDay trả ra giá trị ngày trong tuần tương ứng.

• In kết quả ra màn hình.

3.5 YC5: Cho biết ngày vừa nhập là ngày thứ mấy kể từ ngày 1/1/1.

• Sao chép TIME vào $a0.

• Gọi hàm FullDate.

• Hàm FullDate trả ra kết quả là ngày thứ mấy kể từ ngày 1/1/1.

• In kết quả ra màn hình.

3.6 YC6: Cho biết can chi của năm vừa nhập.

• Sao chép TIME vào $a0.

• Gọi hàm Can, Chi để lấy Can và Chi của năm đó.

• In ra màn hình kết quả.

3.7 YC7: Cho biết khoảng thời gian giữa chuỗi TIME1 và TIME2

• Đọc từ file và lưu TIME2.

• Gọi hàm DateDiff.

• Kết quả hàm DateDiff trả về số ngày cách nhau giữa 2 TIME1 và TIME 2.

• In kết quả ra màn hình.

3.8 YC8: Cho biết 2 năm nhuận gần nhất với năm trong chuỗi TIME.

• Sao chép TIME vào $a0.

• Gọi hàm TwoNearestLeapYear.

• Kết quả từ hàm TwoNearestLeapYear trả ra 2 năm nhuận gần nhất. với năm trong chuỗi TIME.

• In kết quả ra màn hình.

3.9 YC9: Nhập input từ file input.txt xuất kết quả toàn bộ các chức năng trên ra file output.txt

- Gọi hàm DocFile, lưu desciptor của file vào buffer FileWords rồi lưu vào $a1.
- Gọi hàm loadFile để đọc từng ký tự 1 (lưu trong $a1), đọc từng dòng lưu vào $s5 sau đó gọi hàm doitime để đọc tiếp dòng mới. Đến cuối file thì gọi hàm ketthucfile.
- Gọi các hàm kiểm tra định dạng ngày tháng, nếu đúng thì thực hiện bước tiếp theo nếu sai thì bỏ qua
- Hàm Out_File: thực hiện các yêu cầu ở 8 câu trên bằng cách gọi hàm và trả kết quả về $a1, gọi hàm Write_File để lưu các giá trị trong $a1 vào file output.
- Không còn gì để đọc thì kết thúc.

4. Qui tắc khi viết và gọi hàm trong MIPS

a. Trình tự khi viết và gọi hàm

1. Lưu trữ các biến vào nơi mà hàm có thể truy cập đến (truyền tham số cho hàm)
2. Gọi hàm bằng lên **"jal"** (bắt đầu lời gọi hàm)
3. Cấu trúc của một hàm (thủ tục)

*Với mỗi lần gọi hàm bằng "jal", địa chỉ lúc gọi hàm sẽ lưu vào thanh ghi $ra. Do vậy, để tránh việc mất địa chỉ thanh ghi (khi chúng ta gọi nhiều hàm lồng nhau), cần lưu trữ $ra vào stack. Đồng thời cần lưu một số thanh ghi khác: như thanh ghi cho các tham số, hay thanh ghi tạm giữ giá trị kết quả trả về (nếu cần). Vì khi gọi một hàm khác chúng ta có thể làm thay đổi giá trị của thanh ghi đó.

b. Một số nguyên tắc sử dụng thanh ghi

$a0 - $a3	Thanh ghi để lưu các tham số của hàm, có thể thay đổi được. Nên lưu các thanh ghi này vào stack vì thanh ghi có thể thay đổi được
$v0 - $v1	Thanh ghi giữ giá trị trả về của hàm
$t0 - $t9	Thanh ghi lưu giá trị tạm nên có thể thay đổi bất cứ lúc nào. Thủ tục gọi cần lưu lại giá trị nếu nó cần sau các lời gọi thủ tục.
$s0 - $s7	Thanh ghi lưu trữ. Khôi phục nếu thay đổi. Rất quan trọng, vì nếu thủ tục được gọi thay đổi các thanh ghi này thì nó phải phục hồi các thanh ghi trước khi kết thúc.
$ra	Thanh ghi dùng để lưu vị trí của lời gọi hàm và có thể thay đổi được. Thủ tục gọi lưu lại thanh ghi này vào ngăn xếp nếu cần
$sp	Thanh ghi con trỏ ngăn xếp phải có giá trị không đổi trước và sau khi gọi lệnh jal, nếu không thủ tục gọi sẽ không quay về được. Nên cần được khôi phục nếu thay đổi.

5. Tài liệu tham khảo

[1]. Hướng dẫn đồ án 2, project02_Huongdan.pdf, giảng viên Phạm Tuấn Sơn

[2] Slide bài giảng trên moodle Lớp KTMT&HN 17CTT6

[3] https://daynhauhoc.com/t/thao-luan-c-tinh-khoang-thoi-gian-giua-2-thoi-diem-ngay-thang-nam/23812/4 (cho yêu cầu 5 và 7)

[4]. http://mathforum.org/library/drmath/view/66535.html

II. Đánh giá thành viên

MSSV	Họ và tên	Đánh giá hoàn thành
1612087	Vũ Thành Đạt	Tốt
1712173	Đặng Thái Gia Thuận	Tốt
1712805	Lê Đức Thuận	Tốt
1712828	Huỳnh Thanh Khải Trân	Tốt
1712929	Nguyễn Phượng Vỹ	Tốt