TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

MÔN HỌC: KIẾN TRÚC MÁY TÍNH VÀ HỢP NGỮ
LỚP CỬ NHÂN TÀI NĂNG 2017

BÁO CÁO ĐỒ ÁN 2

LẬP TRÌNH MIPS - THƯ VIỆN TIME

Giảng viên:
ThS. Phạm Tuấn Sơn
ThS. Lê Viết Long

Nhóm thực hiện:
1712152 | Nguyễn Thị Mai Thanh
1712228 | Phạm Việt Nga
1712807 | Nguyễn Thị Minh Thùy

MỤC LỤC

1. YÊU CẦU ĐỒ ÁN

Cài đặt thư viện TIME bằng hợp ngữ MIPS với một số thao tác xử lý trên chuỗi ngày tháng.

2. THÔNG TIN THÀNH VIÊN

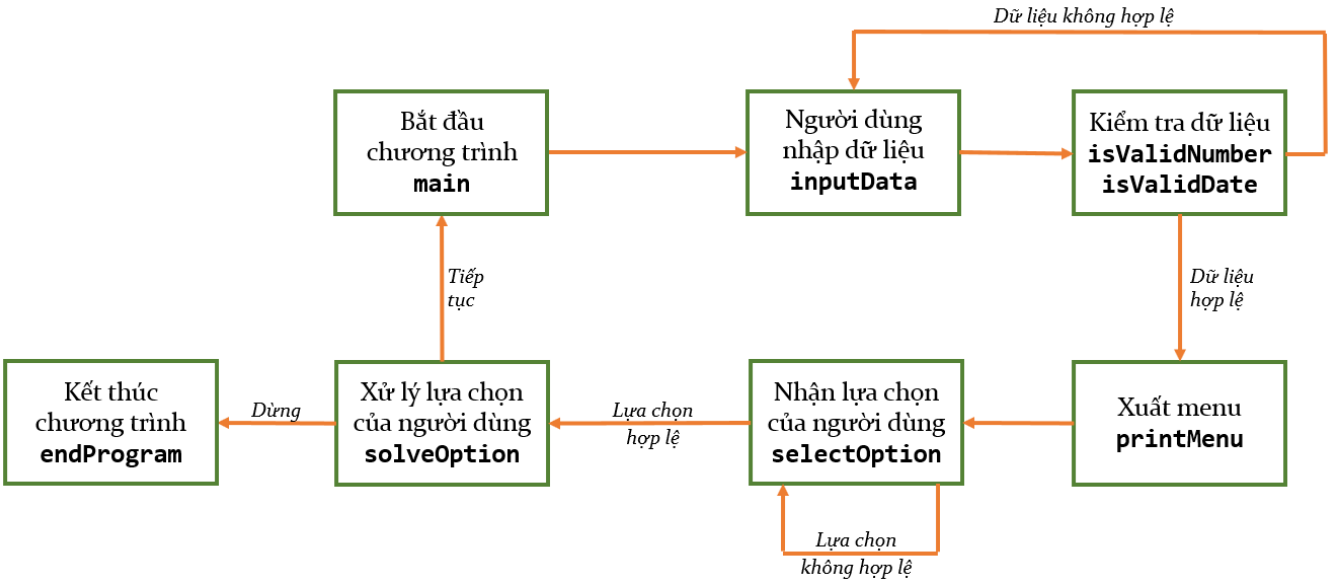
Thành viên	MSSV	Mức độ đóng góp
Nguyễn Thị Mai Thanh	1712152	33.(3) %
Phạm Việt Nga	1712228	33.(3) %
Nguyễn Thị Minh Thùy	1712807	33.(3) %

3. ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH

Yêu cầu	Hoàn thành
Đọc dữ liệu ngày tháng năm do người dùng nhập vào	CÓ
Xuất ra menu cho người dùng lựa chọn thao tác	CÓ
Xuất chuỗi TIME theo định dạng DD/MM/YYYY	CÓ
Chuyển đổi chuỗi TIME sang các định dạng khác	CÓ
Cho biết ngày tháng năm vừa nhập là thứ mấy trong tuần	CÓ
Kiểm tra năm nhuận	CÓ
Cho biết khoảng cách giữa hai chuỗi TIME	CÓ
Cho biết hai năm nhuận gần nhất với năm trong chuỗi TIME	CÓ
Kiểm tra dữ liệu khi nhập	CÓ
Lấy các giá trị ngày, tháng, năm (kiểu int) từ chuỗi TIME	CÓ
Mức độ hoàn thành đồ án:	**100%**

4. CÀI ĐẶT CHƯƠNG TRÌNH

Sơ đồ các hàm chính của chương trình:



5. CÁC HÀM CHỨC NĂNG

5.1. Nhập dữ liệu từ người dùng và lưu vào chuỗi TIME

- Hàm thực hiện: **`inputData`**
  - Tham số: $*a*0 là con trỏ trỏ tới chuỗi TIME, $*a*1 là chuỗi inputString để lưu giá trị người dùng nhập vào.
  - Kết quả: $*v*0 là con trỏ trỏ tới chuỗi TIME ở dạng chuẩn DD/MM/YYYY (trình bày ở 5.3) và $*v*1 là tính hợp lệ của chuỗi TIME.
- Nếu chuỗi hợp lệ, lưu chuỗi TIME vào $*s*0 và nhảy tới `printMenu` và thực hiện các công việc tiếp theo.
- Nếu chuỗi không hợp lệ, nhảy tới `input_again` để in ra thông báo dữ liệu không hợp lệ và yêu cầu nhập lại. Sau đó, nhảy lại về `main` để bắt đầu nhập lại dữ liệu và xử lý (coi như bắt đầu lại chương trình).

5.2. Kiểm tra tính hợp lệ của dữ liệu

Dữ liệu được coi là hợp lệ khi thỏa mãn cả hai điều kiện sau:

- Dữ liệu ngày, tháng, năm nhập vào là chuỗi chỉ chứa các ký tự chữ số.
  - Hàm thực hiện: **`isValidNumber`**
    - Tham số: $*a*0 là chuỗi cần kiểm tra

- Kết quả: $\$v0 = 1/0$ tương ứng với true/ false là tính hợp lệ của chuỗi.

- Thực hiện kiểm tra bằng cách duyệt từng ký tự trong chuỗi xem có phải là chữ số không. Điều kiện được coi là chữ số: ký tự >= 48 và <= 59 (lần lượt là mã ASCII của 0 và 9). Nếu $\$v0 = 1$, chuỗi chỉ toàn các ký tự chữ số → hợp lệ.

- Ngày được biểu diễn phải là ngày hợp lệ:
  - Hàm thực hiện: **`isValidNumber`**
    - Tham số: $\$a0$, $\$a1$, $\$a2$ là các số nguyên (int) tương ứng với ngày, tháng, năm được biểu diễn trong chuỗi TIME
    - Kết quả: $\$v0 = 1/0$ tương ứng với true/ false là tính hợp lệ của ngày tháng năm.
  - Hàm thực hiện kiểm tra trên các điều kiện sau:
    - Tháng phải nằm trong khoảng từ 1 tới 12
    - Năm phải lớn hơn hoặc bằng 1900 (theo yêu cầu trong đề bài) và nhỏ hơn 9999
    - Ngày: phải lớn hơn 0 và:
      - Nhỏ hơn hoặc bằng 31 nếu tháng là $1, 3, 5, 7, 8, 10, 12$
      - Nhỏ hơn hoặc bằng 30 nếu tháng là $4, 6, 9, 11$
      - Nếu tháng là 2: ngày phải nhỏ hơn hoặc bằng 29 nếu năm là năm nhuận, nhỏ hơn hoặc bằng 28 nếu năm không phải là năm nhuận.

    Nếu $\$v0 = 1$ tức là tất cả các điều kiện trên đều thỏa → ngày tháng năm hợp lệ.

5.3. Xuất chuỗi TIME theo định dạng DD/MM/YYYY

- Chuyển 3 chuỗi ngày, tháng, năm do người dùng nhập vào sang chuỗi TIME dạng chuẩn DD/MM/YYYY:
  - 3 chuỗi ngày, tháng, năm do người dùng nhập được chuyển thành số nguyên bằng hàm `toINT`, sau đó 3 số nguyên biểu diễn ngày, tháng, năm này được truyền vào hàm `ConvertToTIME`. Do tính hợp lệ về syntax được kiểm tra trước khi vào hàm nên 3 chuỗi ngày, tháng, năm chỉ gồm các ký tự chữ số.
  - Hàm thực hiện: **`ConvertToTIME`**
    - Tham số: $\$a0$, $\$a1$, $\$a2$ lần lượt là 3 số nguyên biểu diễn ngày, tháng, năm $(day, month, year)$, $\$a3$ là con trỏ trỏ tới chuỗi TIME

- Kết quả: $\$v0$ là con trỏ trỏ tới chuỗi TIME ($\$v0$ được gán bằng $\$a3$ sau khi chuyển đổi).

- Trước khi thực hiện chuyển đổi, hàm `isValidDate` được thực hiện để kiểm tra tính hợp lệ ngày tháng, nếu ngày tháng không hợp lệ thì thoát khỏi hàm, không thực hiện chuyển đổi nữa.
- Chuỗi TIME: DD/MM/YYYY. Các ký tự của chuỗi được xác định bằng công thức:
  - DD: $TIME[0] = day \; div \; 10;$
    $TIME[1] = day \; \% \; 10$
  - MM: $TIME[3] = month \; div \; 10;$
    $TIME[4] = month \; \% \; 10$
  - YYYY: $TIME[6] = year \; div \; 1000;$
    $TIME[7] = (year \; \% \; 1000) \; div \; 100;$
    $TIME[8] = (year \; \% \; 100) \; div \; 10;$
    $TIME[9] = year\%10$
  - $TIME[2] = TIME[5] = TIME[10] = '/'$

- Xuất chuỗi TIME theo dạng chuẩn: Vì chuỗi TIME được lưu ở dạng chuẩn nên chỉ cần gọi syscall 4 để xuất chuỗi ra màn hình.

5.4. Chuyển đổi chuỗi TIME sang các định dạng khác

- Hàm thực hiện: **`Convert`**
  - Tham số: $\$a0$ là chuỗi TIME theo định dạng chuẩn DD/MM/YYYY.
  - Kết quả: $\$v0$ là kết quả chuyển đổi, $\$v1 = 1/0$ tương ứng với true/ false (đã chuyển đổi/ không chuyển đổi do người dùng nhập yêu cầu sai).
- Hàm `Convert` thực hiện chức năng nhận yêu cầu của người dùng, xử lý tính hợp lệ của lựa chọn chức năng và gọi hàm con tương ứng thực hiện chức năng đó.
  - Input 'A': goi hàm **`convert_according_to_A`** (MM/DD/YYYY).
  - Input 'B': goi hàm **`convert_according_to_B`** (Month DD, YYYY).
  - Input 'C': goi hàm **`convert_according_to_C`** (DD Month, YYYY).
  - Input khác ba trường hợp trên, thông báo không hợp lệ bằng hàm `Convert_invalid`.

A. MM/DD/YYYY

Định dạng kiểu A chính là hoán đổi DD và MM với nhau trong chuỗi chuẩn TIME. Do đó, chỉ cần lấy ra 4 vị trí của 4 ký tự chứa 2 thành phần đó và hoán đổi thích hợp.

$$DD = TIME[0] + TIME[1] = 0(\$a0) + 1(\$a0)$$
$$MM = TIME[3] + TIME[4] = 3(\$a0) + 4(\$a0)$$

B. Month DD, YYYY

Thực hiện chuyển đổi bằng cách lấy ra từng thành phần của chuỗi TIME, sau nối lại thành một chuỗi mới theo định dạng B.

- Lấy số nguyên biểu diễn tháng bằng hàm `Month`. Sau đó, chuyển tháng thành tên tương ứng với hàm `NameOfMonth` (1 ~ January, …, 12 ~ December).
- Sao chép 'DD' từ TIME qua chuỗi *tmp*_1.
- Lưu 'YYYY' từ TIME vào chuỗi *tmp*_2 (sao chép từng kí tự), các kí tự bổ sung theo định dạng dựa theo bảng ASCII.
- Nối chuỗi bằng hàm `strcpy` và `strcat` để tạo thành chuỗi theo định dạng yêu cầu: `Month` + *tmp*_1 + *tmp*_2

C. DD Month, YYYY

Tương tự như định dạng B, thứ tự nối chuỗi sẽ được đảo vị trí cho phù hợp với định dạng C: *tmp*_1 + `Month` + *tmp*_2

5.5. Cho biết ngày trong chuỗi TIME là thứ mấy trong tuần

- Hàm thực hiện: **`WeekDay`**
  - Sử dụng các hàm `Day`, `Month`, `Year` để lấy ngày ($\$a1$), tháng ($\$a2$), năm($\$a3$) trong chuỗi TIME ban đầu ($\$a0$).
  - Kiểm tra điều kiện với $\$a2$ để thực hiện việc tính toán và gọi hàm `indentify_WeekDay`.
- Hàm **`indentify_WeekDay`**: tính toán theo công thức
  - Công thức: $n = (d + 2 * m + 3 * (m + 1)\ div\ 5 + y + y\ div\ 4)\ mod\ 7$
    Với d: ngày, m: tháng, y: năm, n: thứ.
  - Điều kiện: nếu $m < 3$ thì thực hiện tính toán
    - ✓ $m = m + 12$
    - ✓ $y = y - 1$

    Sau đó, vẫn sử dụng công thức như trên.
  - Kết quả n ($\$v0$) thuộc tập giá trị $\{0, 1, 2, 3, 4, 5, 6\}$ ứng với thứ trong tuần {chủ nhật, thứ 2, thứ 3, thứ 4, thứ 5, thứ 6, thứ 7}.

5.6. Kiểm tra năm trong chuỗi TIME có phải là năm nhuận hay không

- Hàm chính **`LeapYear`**:
  - Tham số: $a0 là chuỗi TIME theo định dạng chuẩn DD/MM/YYYY.
  - Kết quả: $v0 = 1/ 0 (Năm nhuận/ Không phải năm nhuận)
- Các bước thực hiện:
  - Lấy giá trị năm trong chuỗi TIME (bằng hàm `Year`).
  - Dùng hàm `OnlyYear_LeapYear` để trả về kết quả 1 (năm nhuận) hoặc 0 (không phải năm nhuận)
- Hàm phụ **`OnlyYear_LeapYear`**:
  - Tham số: $a0 là số nguyên biểu diễn năm.
  - Kết quả: $v0 là 1 nếu năm cần kiểm tra là năm nhuận hoặc 0 nếu năm cần kiểm tra không phải năm nhuận
- Điều kiện kiểm tra năm nhuận: năm chia hết cho 400 hay năm chia hết cho 4 nhưng không chia hết cho 100 được coi là năm nhuận.
- Các bước kiểm tra:
  - Xét nếu năm chia hết cho 400 → trả về 1 (năm nhuận), ngược lại tiếp tục kiểm tra.
  - Xét nếu năm không chia hết cho 4 → trả về 0 (không phải năm nhuận), ngược lại tiếp tục kiểm tra.
  - Xét nếu năm chia hết cho 100 (chia hết cho 4 và cho 100 nhưng không chia hết cho 400) → trả về 0 (không phải năm nhuận). Ngược lại, năm không chia hết cho 100 (chia hết cho 4 nhưng không chia hết cho 100) → trả về 1 (năm nhuận).
- Ví dụ:
  - 2000 là năm nhuận
  - 1800 không phải năm nhuận (1800 chia hết cho 4, 100 nhưng không chia hết cho 400)

5.7. Cho biết khoảng cách giữa hai chuỗi TIME (tính theo năm)

- Hàm chính **`GetTime`**:
  - Tham số: $a0, $a1 là 2 chuỗi TIME theo định dạng DD/MM/YYYY
  - Kết quả: $v0 là số nguyên chỉ khoảng cách giữa 2 chuỗi TIME (tính theo năm)
- Các bước thực hiện:
  - So sánh chuỗi *TIME*_1 và *TIME*_2 để tìm ra chuỗi lớn hơn (sử dụng hàm `CompareTime`).

- Xét nếu $TIME\_1 = TIME\_2$ thì xuất khoảng cách năm là 0.
- Sau đó sử dụng hàm `SubYear` để tính khoảng cách giữa 2 chuỗi $TIME\_1$ và $TIME\_2$ với $\$a0$ là chuỗi TIME lớn hơn, $\$a1$ là chuỗi TIME nhỏ hơn.

- Các hàm phụ trợ:
  - **`CompareTime`**:
    - Tham số: $\$a0, \$a1$ là 2 chuỗi theo thứ tự $TIME\_1$ và $TIME\_2$
    - Kết quả: $vo mang 1 trong các giá trị: 0 ($TIME_1 = TIME_2$), 1 ($TIME_1 > TIME_2$), -1 ($TIME\_1 < TIME\_2$)
    - Hàm thực hiện tách 2 chuỗi TIME ra 3 số nguyên tương ứng với ngày, tháng, năm (bằng các hàm `Day`, `Month`, `Year`) và thực hiện so sánh theo thứ tự năm -> tháng -> ngày.
  - **`SubYear`**:
    - Tham số: $\$a0, \$a1$ là 2 chuỗi TIME theo định dạng DD/MM/YYYY
    - Kết quả: $\$v0$ là số nguyên chỉ khoảng cách giữa 2 chuỗi TIME (tính theo năm)
    - Các bước thực hiện:
      - Tách 2 chuỗi TIME ra 3 số nguyên tương ứng với ngày, tháng, năm (bằng các hàm `Day`, `Month`, `Year`).
      - Xét năm: nếu năm bằng nhau → trả về 0, ngược lại: $\$v0 = year(TIME\_1) - year(TIME\_2)$ (do $TIME\_1 > TIME\_2$)
      - Xét tháng:
        - Nếu $month(TIME\_1) > month(TIME\_2)$ → trả về $\$v0$ hiện tại.
        - Nếu $month(TIME\_1) < month(TIME\_2)$ → chưa đủ 1 năm → trả về $\$v0 - 1$.
        - Nếu $month(TIME\_1) = month(TIME\_2)$ → xét ngày
      - Xét ngày:
        - Nếu $day(TIME_1) \geq day(TIME\_2)$ → trả về $\$v0$ hiện tại.
        - Nếu $day(TIME\_1) < day(TIME\_2)$ → chưa đủ 1 năm → trả về $\$v0 - 1$.

5.8. Cho biết hai năm nhuận gần nhất với năm trong chuỗi TIME

- Hàm thực hiện: **`LeapYearCloser`**
  - Tham số: $\$a0$ là chuỗi TIME theo định dạng chuẩn DD/MM/YYYY

- Kết quả: $\$v0, \$v1$ là 2 năm nhuận gần nhất

- Các bước thực hiện:
  - Lấy giá trị năm trong chuỗi TIME (bằng hàm `Year`) lưu vào $\$t0$
  - Tìm năm nhuận lớn hơn năm hiện tại (`LeapYearCloser_Next`): Dùng $\$t1$ để lưu khoảng cách từ năm hiện tại đến năm nhuận lớn hơn gần nhất. Cộng 1 liên tục vào $\$t1$ cho đến khi tìm được $\$t0 + \$t1$ là năm nhuận.
  - Tìm năm nhuận nhỏ hơn năm hiện tại (`LeapYearCloser_Past`): Dùng $\$t2$ để lưu khoảng cách từ năm hiện tại đến năm nhuận nhỏ hơn gần nhất. Cộng 1 liên tục vào $\$t2$ cho đến khi tìm được $\$t0 - \$t2$ là năm nhuận nhưng năm tìm được phải lớn hơn 1900.
  - So sánh khoảng cách năm nhuận gần nhất so với năm hiện tại:
    + Tìm năm nhuận kế tiếp của năm nhuận (lớn hơn năm hiện tại) vừa tìm được. Dùng $\$t4$ để lưu khoảng cách
    → $\$t4 = distance(current\_year, year\_next\_next)$
    + Tìm năm nhuận kế tiếp của năm nhuận (nhỏ hơn năm hiện tại) vừa tìm được. Dùng $\$t5$ để lưu khoảng cách
    → $\$t5 = distance(year\_past\_past, current\_year)$
    + Tính khoảng cách $distance(year\_past, year\_next) = \$t2 + \$t1$
    + Tìm khoảng cách nhỏ nhất trong 3 kết quả trên (dùng hàm Min) và trả về kết quả:
    - Nếu $\$t4$ nhỏ nhất => $\$v0 = \$t0 + \$t1$ và $\$v1 = \$t0 + \$t4$
    - Nếu $\$t5$ nhỏ nhất => $\$v0 = \$t0 - \$t5$ và $\$v1 = \$t0 - \$t2$
    - Nếu $\$t1 + \$t2$ nhỏ nhất => $\$v0 = \$t0 - \$t2$ và $\$v1 = \$t0 + \$t1$
  - Nếu năm nhuận nhỏ hơn cần tìm < 1900 → Tìm năm nhuận kế tiếp của năm nhuận vừa tìm được ở hàm `LeapYearCloser_Next` bằng hàm `LeapYearCloser_Next_Next` bằng cách cộng liên tục 4 vào kết quả cho đến khi tìm được năm nhuận.

- Ví dụ:
  - Ngày hiện tại 21/02/1999 → Hai năm nhuận gần nhất là 1996 và 2000
  - Ngày hiện tại 21/02/1900 → Hai năm nhuận gần nhất là 1904 và 1908
  - Ngày hiện tại 21/02/2103 → Hai năm nhuận gần nhất là 2104 và 2108 (do 2100 không phải năm nhuận, mà năm nhuận nhỏ hơn cần tìm là 2096 → chưa phải hai năm nhuận gần nhất → chọn hai năm nhuận lớn hơn là 2104 và 2108)

- Ngày hiện tại 21/02/2098 → Hai năm nhuận gần nhất là 2092 và 2096 (do 2100 không phải năm nhuận, mà năm nhuận lớn hơn cần tìm là 2104 → chưa phải hai năm nhuận gần nhất → chọn hai năm nhuận lớn hơn là 2092 và 2096)

6. TÀI LIỆU THAM KHẢO

- Bài giảng **Kiến trúc MIPS** – ThS. Phạm Tuấn Sơn, ĐH Khoa học Tự nhiên, ĐHQG TP.HCM
- **MIPS Assembly/MIPS Instructions** https://en.wikibooks.org/wiki/MIPS_Assembly/MIPS_Instructions