TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
MÔN: KIẾN TRÚC MÁY TÍNH VÀ HỢP NGỮ
LỚP: 16CNTN

BÁO CÁO ĐỒ ÁN 2

THƯ VIỆN TIME

Người thực hiện:
Nguyễn Quốc Vương – 1612829
Nguyễn Thanh Tuấn – 1612774
Lê Thành Công – 1612842

Giáo viên:
Phạm Tuấn Sơn

Mục lục

I. Nội dung.

1. Đánh giá mức độ hoàn thành.

Đánh giá tổng quan: Hoàn thành tất cả các yêu cầu của độ án (100%).

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	Cho biết ngày vừa nhập là ngày thứ mấy trong tuần		100	
4	Kiểm tra năm trong chuỗi TIME có phải là năm nhuận không		100	
5	Cho biết khoảng thời gian giữa chuỗi TIME_1 và TIME_2		100	
6	Cho biết 2 năm nhuận gần nhất với năm trong chuỗi time		100	
7	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 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 inputDay: Nhập chuỗi ngày.

- Nhãn loopDay: 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 outDay.
- Nhãn errorDay: Xuất chuỗi báo lỗi, yêu cầu nhập lại và quay trở lại nhãn inputDay để nhập lại chuỗi ngày.
- Nhãn outDay: Chuyển chuỗi vừa nhập kiểu char* thành dạng int.
- Nhãn inputMonth: Nhập chuỗi tháng.
- Nhãn loopMonth: 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 outMonth.
- Nhãn errorMonth: Xuất chuỗi báo lỗi, yêu cầu nhập lại, quay trở lại nhãn inputMonth để nhập lại chuỗi tháng.
- Nhãn outMonth: Chuyển chuỗi vừa nhập kiểu char* sang int.
- Nhãn inputYear: Nhập chuỗi năm.
- Nhãn loopYear: 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 outYear.
- Nhãn errorMonth: Xuất chuỗi báo lỗi, yêu cầu nhập lại, quay trở lại nhãn inputYear để nhập lại chuỗi năm.
- Nhãn outYear: 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 errorDay – 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.

2.2. 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: Trả về giá trị mà biến TIME đang giữ.
- Thuật toán, chi tiết xử lý các bước:

  - Lưu các biến tạm sẽ sử dụng vào stack.
  - Chuyển ngày thành chuỗi, gắn thêm kí tự ‘/’ vào cuối chuỗi (được chuỗi “DD/”), chuyển tháng thành chuỗi gắn vào tiếp với chuỗi ngày, gắn thêm kí tự ‘/’ vào cuối chuỗi (được chuỗi “DD/MM/”), chuyển năm thành chuỗi gắn với chuỗi trước đó, ta được chuỗi cuối cùng “DD/MM/YYYY”.
  - Nhãn loopDate: Lần lượt đưa các kí tự vào chuỗi TIME.
  - Nhãn outDate: Thêm kí tự kết thúc chuỗi vào chuỗi TIME.

- Nhãn endDate: Trả lại các biến tạm lưu vào stack, trả về.

2.3. 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.
- 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_checkDate: Trả lại các biến tạm lưu vào stack, trả về.

2.4. Hàm int numDayOfMonth(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 endNumDayOfMonth: Trả lại các biến tạm lưu vào stack, trả về.

2.5. 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.

2.6. 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 *trueLeapYear*, ngược lại, nếu năm truyền vào chia hết cho 100, nhảy tới nhãn *falseLeapYear;* ngược lại, nếu năm truyền vào không chia hết cho 4, nhảy tới nhãn *falseLeapYear*.
  - Nhãn trueLeapYear: Gán kết quả bằng 1, nhảy đến nhãn end_isLeapYear.
  - Nhãn falseLeapYear: Gán kết ủa bằng 0.
  - Nhãn: end_isLeapYear: Trả các biến đã lưu vào stack, trả về.

2.7. 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.

2.8. 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 *endTextMonth.*
  - Nhãn: end_textMonth: Trả các biến đã lưu vào stack, trả về.

2.9. 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 tiếng Anh 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 *cal_Weekday*. Ngược lại, tháng=tháng+12, năm=năm-1.
  - Nhãn cal_WeekDay:
    - 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 *sunday, monday, tuesday, wednesday, thursday, friday, saturday*.
    - Với lần lượt các nhãn trên, thì kết quả trả về là con trỏ trỏ đến vùng nhớ chứa chuỗi tương ứng thuộc tập. {“Sunday”, ”Monday”, ”Tuesday”, ”Wednesday”, ”Thursday”, ”Friday”, ”Saturday”} và nhảy tới nhãn *end_Weekday.*
  - Nhãn end_WeekDay: Trả các biến đã lưu vào stack, trả về.

2.10. 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.
    - Biến đếm tạm count=0.
  - Nhãn *loop_twoNearestLeapYear*:
    - Năm=năm+1.
    - Truyền mặc định ngày 1 tháng 1 và Năm kết quả hiện hành cùng với chuỗi tempYear để lưu năm dạng chuỗi và gọi hàm Date ta nhận được kết quả là dạng chuỗi của ngày tháng năm.
    - 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 *print_nearestLeapYear.*
    - Tiếp tục vòng lặp bằng cách nhảy lại tới nhãn *loop_twoNearestLeapYear.*
  - Nhãn *print_nearestLeapYear:*
    - Count=count+1.
    - Nếu count==3 nhảy tới nhãn *end_twoNearestLeapYear*.
    - In giá trị năm nhuận ra màn hình.
    - In khoảng trắng
    - Nhảy tới nhãn *loop_twoNearestLeapYear.*
  - Nhãn end_ *twoNearestLeapYear*: Trả các biến đã lưu vào stack, trả về.

2.11. Hàm int GetTime (char* TIME1, char* TIME2).

- Mô tả: Tính khoảng thời gian cách biệt giữa giá trị năm 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 năm cách biệt.
- Thuật toán, chi tiết xử lý các bước:

  - Nhãn *GetTime*:

- Lưu các biến tạm sẽ sử dụng vào stack.
- Biến đếm tạm count=0.
- Với TIME1 và TIME2 là đầu vào để gọi hàm Year để lấy giá trị nguyên của năm trong chuỗi và lưu vào các biến tạm.
- Nếu năm_1-năm_2==0 thì nhảy tới nhãn *end_GetTime.*
- Count=năm_1-năm_2.
- Nếu năm_1<năm_2 thì nhảy tới nhãn *if_GetTime.*
- Dùng hàm *Month* lần lượt lấy ra giá trị nguyên của tháng trong chuỗi TIME_1 và TIME_2.
- Nếu tháng_1<tháng_2 thì nhảy tới nhãn *count_sub_1*.
- Nếu tháng_1>tháng_2 thì nhảy tới nhãn *end_GetTime.*
- Ngược lại, nếu tháng_1==tháng_2 thì dùng hàm *Day* lần lượt lấy ra giá trị nguyên của ngày trong TIME_1 và TIME_2.
- Nếu ngày_1<ngày_2 thì nhảy tới nhãn *count_sub_1*.
- Nhảy tới nhãn *end_GetTime.*

- Nhãn if_GetTime:
  - Count=năm_2-năm_1.
  - Dùng hàm *Month* lần lượt lấy ra giá trị nguyên của tháng trong chuỗi TIME_1 và TIME_2.
  - Nếu tháng_2<tháng_1 thì nhảy tới nhãn *count_sub_1*.
  - Nếu tháng_2>tháng_1 thì nhảy tới nhãn *end_GetTime.*
  - Ngược lại, nếu tháng_2==tháng_1 thì dùng hàm *Day* lần lượt lấy ra giá trị nguyên của ngày trong TIME_1 và TIME_2.
  - Nếu ngày_2<ngày_1 thì nhảy tới nhãn *count_sub_1*.
  - Nhảy tới nhãn *end_GetTime.*
- Nhãn count_sub_1: Count = Count – 1.
- Nhãn *end_GetTime*: Trả các biến đã lưu vào stack, trả về.

2.12. 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.

2.13. 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.

2.14. 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.

2.15. 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.

2.16. Hàm char * itoa(char*stringNum, int num).

- Mô tả: hàm chuyển số nguyên thành chuỗi số.
- Input: chuỗi stringNum rỗng (kiểu char*) và số nguyên cần chuyển (kiểu int).
- Output: dạng chuỗi của số đó sau khi chuyển kiểu char*.
- Thuật toán, chi tiết xử lý các bước:

  - lưu các giá trị của các thanh ghi có thể thay đổi vào vùng nhớ stack.
  - gán $t0 = $a0, $t0 lưu địa chỉ của chuỗi stringNum.
  - vòng lặp:
    - nếu $a1 == 0 thì kết thúc.
    - $a1 chia lấy dư cho 10 để lấy kí số ở cuối, cộng với 48 để chuyển kí số đó thành kí tự số, lưu kí tự đó vào vị trí đầu của chuỗi stringNum ($t0), đồng thời chia lấy nguyên cho 10 để lấy các kí số còn lại dùng cho vòng lặp kế tiếp.
    - kết thúc vòng lặp ta được một số dạng chuỗi stringNum ($t0) nhưng nó bị đảo ngược.
      - Cần viết hàm reverse để đảo chuỗi này lại.

- gọi hàm reverse (do $t0 mỗi lần lặp được cộng giá trị nên sau vòng lặp nó lưu địa chỉ ở cuối chuỗi stringNum, trong khi trước đó ta có $a0 lưu giá trị đầu chuỗi stringNum, nên chỉ cần gọi jal reverse (với $a0 là tham số truyền vào lưu địa chỉ đầu chuỗi stringNum) sẽ được $v0 là chuỗi số kết quả.
- trả lại các giá trị đã lưu trong vùng nhớ stack cho các thanh ghi trước đó và kết thúc hàm.

2.17. Hàm reverse(char* stringNum).

- Mô tả: hàm đảo ngược chuỗi.
- Input: chuỗi cần đảo kiểu char*.
- Output: chuỗi đã đảo ngược kiểu char*.
- Thuật toán, chi tiết xử lý các bước:

  - gán $t0 = $a0.
  - vòng lặp 1:
    - duyệt từng kí tự của $t0.
    - nếu đó là kí tự '\0' thì kết thúc vòng lặp.
    - kết thúc vòng lặp $t0 lưu địa chỉ cuối chuỗi stringNum tại vị trí '\0'.
  - $t0 = $t0 – 1 để $t0 lưu địa chỉ ở kí tự cuối chuỗi stringNum.
  - gán $t1 = $a0 để thao tác trên $t1 còn $a0 vẫn lưu địa chỉ đầu chuỗi stringNum.
  - vòng lặp 2:
    - $t1 lưu địa chỉ kí tự đầu chuỗi, $t0 lưu địa chỉ kí tự cuối chuỗi.
    - mỗi lần lặp $t1 tăng 1, $t0 giảm 1 nếu $t1 >= $t0 thì kết thúc.
    - mỗi lần lặp hoán vị giá trị của $t1 và $t0.
    - kết thúc vòng lặp ta được stringNum đã đảo ngược.
  - gán $v0 bằng $a0 vì lúc này $a0 vẫn lưu địa chỉ đầu chuỗi stringNum.

3. Tài liệu tham khảo

[1]. Slide bài giảng 5_Hop_ngu_MIPS.pdf, giảng viên Phạm Tuấn Sơn.
[2]. Hướng dẫn đồ án 2, project02_Huongdan.pdf, giảng viên Phạm Tuấn Sơn.
[3]. Slide bài giảng Bai05_Kien_truc_MIPS.pdf, giang vien Phạm Tuấn Sơn

II. Đánh giá thành viên

MSSV	Họ và tên	Đánh giá hoàn thành
1612829	Nguyễn Quốc Vương	Tốt
1612774	Nguyễn Thành Tuấn	Tốt
1612842	Lê Thành Công	Tốt