BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH

NHÓM SINH VIÊN THỰC HIỆN : NHÓM 12

ĐỀ TÀI 18

Sinh viên thực hiện: Ninh Thị Bích

MSSV: 20176698

Đề bài:

Two arrays are called similar if one can be obtained from another by swapping at most one pair of elements in one of the arrays.

Given two arrays a and b, check whether they are similar. Example:

-For a = [1, 2, 3] and b = [1, 2, 3], the output should be areSimilar(a, b) = true. The arrays are equal, no need to swap any elements.

-For a = [1, 2, 3] and b = [2, 1, 3], the output should be areSimilar(a, b) = true. We can obtain b from a by swapping 2 and 1 in b.

-For a = [1, 2, 2] and b = [2, 1, 1], the output should be areSimilar(a, b) = false. Any swap of any two elements either in a or in b won't make a and b equal.

Nội dung thực hiện:

1. Cách thực hiện.

* Ý tưởng giải quyết bài toán:
* Đề bài yêu cầu kiểm tra xem hai mảng A, B có tương đương không nghĩa là ta có thể hoán vị tối đa các cặp phần tử trong mảng A để tạo ra mảng B. Vì vậy ta có một cách giải quyết sau:

B1: Nhập số lượng phần tử của mảng. ( chỉ yêu cầu nhập một lần khi bắt bầu chạy chương trình vì nếu số lượng phần tử của hai mảng là khác nhau thì ta hiểu chúng là không tương đương rồi, ở đây ta chỉ xét trường hợp hai mảng có số lượng phần tử bằng nhau). Khi thực hiện việc nhập cần kiểm tra xem giá trị nhập vào có là giá trị số hay không. Nếu sai thì yêu cầu nhập lại.

B2: Nhập vào mảng A, B.

B3: Tiến hành sắp xếp hai mảng A,B.

B4: Kiểm tra xem A[i] và B[i] có giống nhau không. Nếu có một vị trí i mà giá trị A[i] khác B[i] thì đưa ra kết quả là FALSE.

* Phân tích các bước:
* B1: Hiển thị ra lời nhắn “Nhap so luong phan tu cua mang:" có nhãn là size thì sử dụng dịch vụ “printf string”. Sử dụng nhãn CheckNumber để kiểm tra giá trị vừa nhập.
* B2: Nhập các phần tử của mảng A,B. Ta sử dụng hàm read\_array để đọc hai mảng. Cần kiểm tra giá trị vừa nhập bằng hàm CheckNumber.
* B3: Tiến hành sắp xếp mảng A và B. Sử dụng hàm sort\_array.
* B4: Sử dụng hàm areSimilar để kiểm tra hai mảng A,B có tương đương không.
* Chương trình chính:

----------------------------------------------------------------

#main()

# nhập vào số lượng phần tử

#read\_array(A) # doc vao mang A

#read\_array(B) # doc vao mang B

# sort\_array(A) # sap xep mang A

#sort\_array(B) # Sap xep mang B

# areSimilar(A,B) # check two arrays are similar ?

-----------------------------------------------------------------

1. Ý nghĩa của các thanh ghi và các hàm (chương trình con) được sử dụng.

* Bước 1:

\* Các thanh ghi $v0, $a0 dùng cho các dịch vụ printf string, read string. Thanh ghi $s6 để lưu giá trị số phần tử của mảng.

\* Sử dụng nhãn CheckNumber. Để thực hiện hàm này ta có các nhãn: “Soam” (kiểm tra số âm) và “Chuyenso” (chuyển từ xâu sang số).

- Ở hàm CheckNumber gồm cả nhãn loopcheck, chuyenso, soam có các thanh ghi sau:

+ $s7: Cờ kiểm soát lỗi: s7=0 thì không có lỗi, s7=1 có lỗi và đưa ra thông báo nhập lại.

+ $t9: vị trí i của kí tự trong xâu.

+$s4: giá trị được chuyển từ xâu sang số

+$s3: cờ báo số âm: s3=0 thì không phải số âm,s3=1 thì là số âm.

+ $t2: chứa Address string[i]

+$t3: chứa giá trị string[i]

+ $s1: gán giá trị ‘0’ cho s1

+ $s2: gán giá trị ‘9’ cho s2

+$t4: string[i] >= '0' ? $t4 = 1 : 0

+$t5: string[i] <= '9' ? $t5 = 1 : 0

+$t1: '0' <= string[i] <= '9' ? $t1 = 1 : 0

+$t8: lấy giá trị số của kí tự

* Bước 2:

\*Sử dụng hàm read\_array để đọc mảng A,B.Trong đó có nhãn loop\_read là vòng lặp để nhập phần tử mảng.Các thanh ghi :

+ $sp: con trỏ stack để lưu địa chỉ của $pc sau khi thực hiện CheckNumber, trở về main.

+ Thanh ghi $s0: địa chỉ của mảng.

+ Thanh ghi $t0 để lưu giá trị i.

+ Thanh ghi $t1 để lưu giá trị 0 khi i>n và lưu giá trị 1 khi i<n.

+ Dịch vụ “read integer” để đọc vào phần tử array[i] , thanh ghi $v0 để lưu số vừa nhập.

+ Thanh ghi $ra trả về địa chỉ trước khi thực hiện hàm read\_array.

+ Ngoài ra vẫn có các thanh ghi $s7,$s4.

* Bước 3:

#---------------mã code C---------------------------

for(int i=0; i<n; i++){

# for(int j=i+1;j<n;j++){

# if(array[i]<array[j]){

# int temp = array[i]; array[i]= array[j]; array[j]=temp;}

#--------------------------------------------------------------------------------------

\*Sử dụng hàm sort\_array để sắp xếp hai mảng A,B. Ở Hàm sử dụng 2 vòng for là FOR\_I và FOR\_J

+ Thanh ghi $t0 để lưu giá trị i.

+ Thanh ghi $t1 để lưu giá trị 0 khi i>n và lưu giá trị 1 khi i<n

+ Thanh ghi $s1 chứa địa chỉ của array[i].

+ Thanh ghi $t2: lưu giá trị 4i.

+ Thanh ghi $t3 : lưu giá trị j; j=i+1

+ Thanh ghi $s4 để load giá trị của array[i]

+ Thanh ghi $s2 chứa địa chỉ của array[j]

+Thanh ghi $t4 : lưu giá trị 4j.

+ Thanh ghi $s5 để load giá trị của array[j]

+Các nhãn nextFORI và nextFORJ để lặp lại vòng lặp for.

+ Thanh ghi $ra : trở về chương trình chính.

* Bước 4:

\*Sử dụng hàm areSimilar để kiểm tra hai mảng có tương đương không.

#----------------Mã C--------------------------------

# for(int i=0; i<n; i++){

# if(i==n) return true;

# if(arrayA[i]!=arrayB[i]) return false ;

# }

#-------------------------------------------------------

+ Thanh ghi $t0 để lưu giá trị i.

+ Thanh ghi $t1 lưu giá trị 4i.

+ Thanh ghi $s4 để load giá trị A[i]

+ Thanh ghi $s5 để load giá trị B[i]

+ Thanh ghi $ra

+ Các nhãn true\_end và false\_end được sử dụng để đưa ra lời ‘’areSimilar(A,B)=true’’ và ‘’areSimilar(A,B)=false" bằng cách sử dụng dịch vụ “MessageDialog”.

ĐỀ TÀI 17:

Sinh viên thực hiện: Ôn Như Hà

MSSV: 20176743

Đề bài:

Write a program that inputs a string. Extract number characters and show to screen in inverse order using stack.

Nội dung thực hiện:

*1. Cách thực hiện:*

- Ý tưởng giải quyết bài toán:

* Bước 1: yêu cầu người dùng nhập vào một xâu.
* Bước 2: duyệt lần lượt các kí tự trong xâu, kiểm tra các kí tự có phải là kí tự số hay không. Nếu không có kí tự số thì in ra thông báo lỗi.
* Bước 3: push lần lượt các kí tự số vào trong stack.
* Bước 4: pop các kí tự số ra khỏi stack rồi in ra kết quả.

-Phân tích các bước:

* Bước 1: hiển thị lời nhắn : “hãy nhập vào một xâu:” yêu cầu người dùng nhập xâu .
* Bước 2: duyệt lần lượt từng kí tự kiểm tra xem kí tự đó có phải kí tự số bằng loop1 và loop2 :
  + - * + Với loop1 là kiểm tra điều kiện thứ nhất kí tự đó >=0.

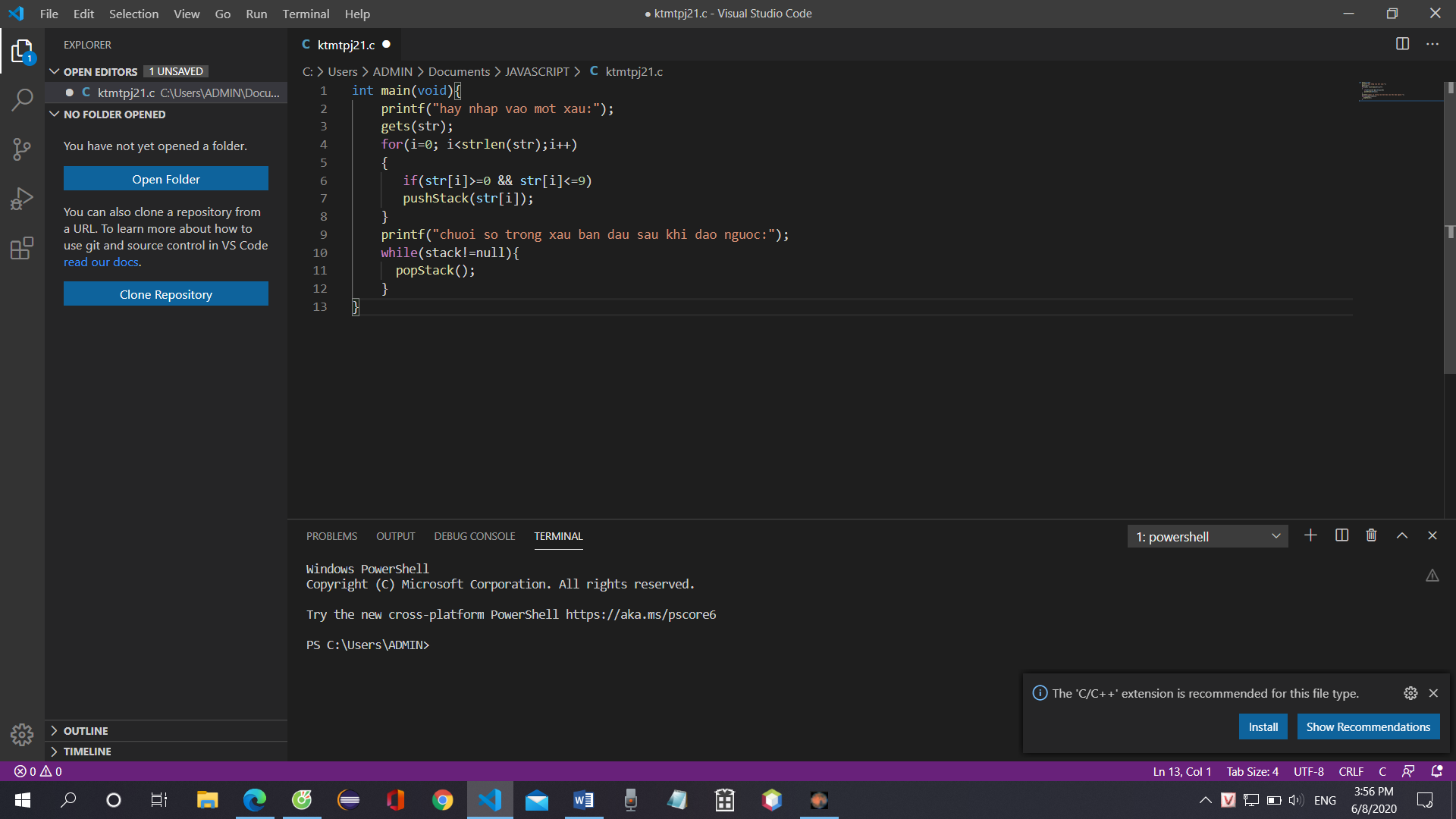
(trong bảng mã ASCII kí tự số 0 có số thứ tự là 48)

* + - * + Với loop2 là kiểm tra điều kiện thứ hai: kí tự đó <= 9.

(trong bảng ma ASCII kí tự số 9 có số thứ tự là 57)

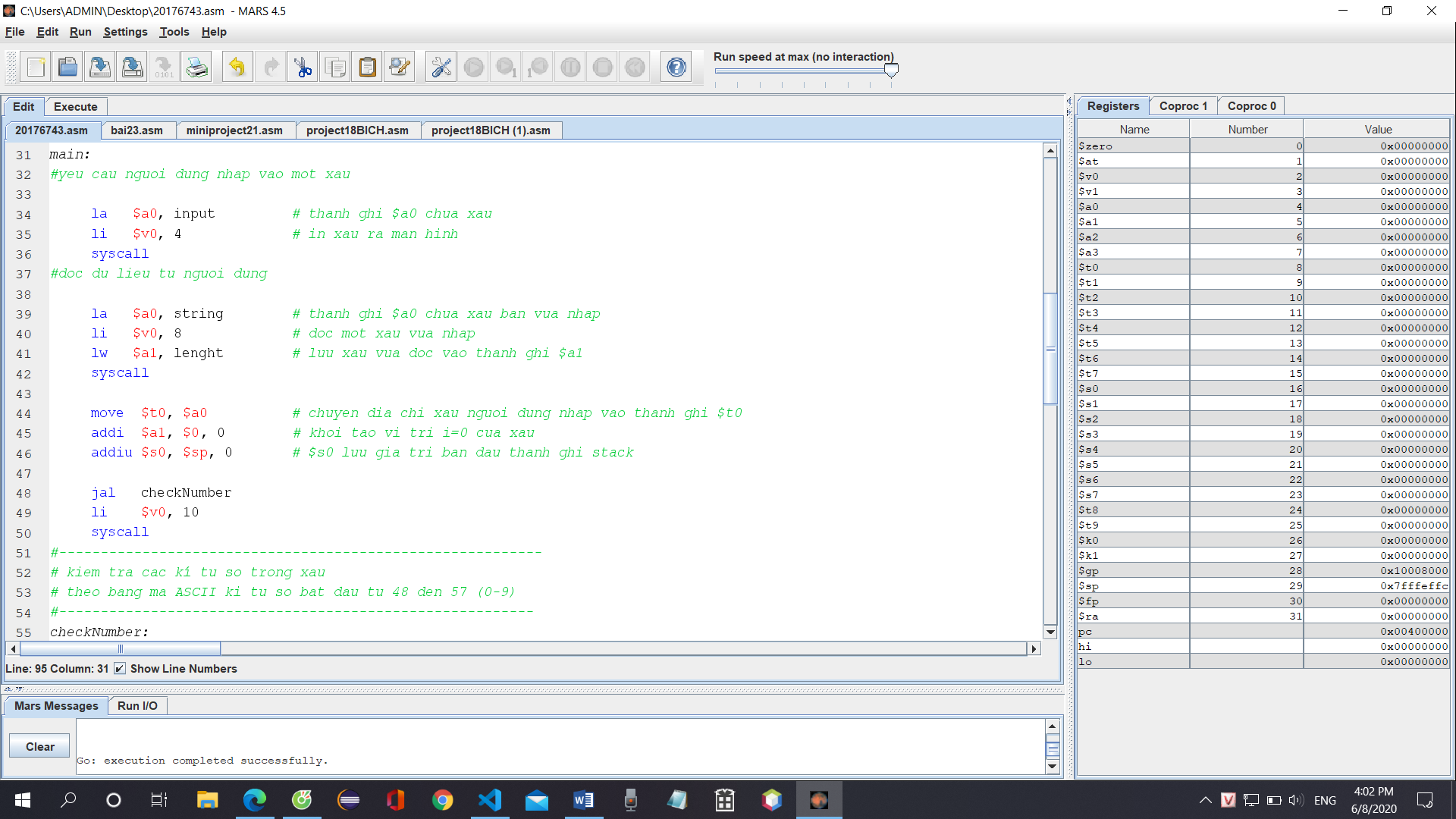
* Bước 3: kí tự sau khi đã thỏa mãn 2 điều kiện 🡪là kí tự số🡪push vào trong stack.
* Bước 4: sau khi đã push tất cả các kí tự số có trong xâu vào trong stack 🡪 pop lần lượt các kí tự ra stack🡪 in các kí tự.
* Giải thích: do stack hoạt động theo nguyên tắc: vào sau ra trước (Last in first out - LIFO) nên khi ta push các kí tự số vào stack thì khi pop các kí tự đó ra khỏi stack và in ra ta sẽ được một chuỗi các kí tự số ngược với chuỗi trước khi ta push vào stack.

Mã C:



*2. Ý nghĩa các thanh ghi và các hàm con được sử dụng.*

* Thanh ghi $t0: chứa địa chỉ xâu của người dùng nhập vào ban đầu.
* Thanh ghi $a1: chứa giá trị thứ i của xâu .
* Thanh ghi $s0: chứa giá trị thanh ghi ban đầu của stack.

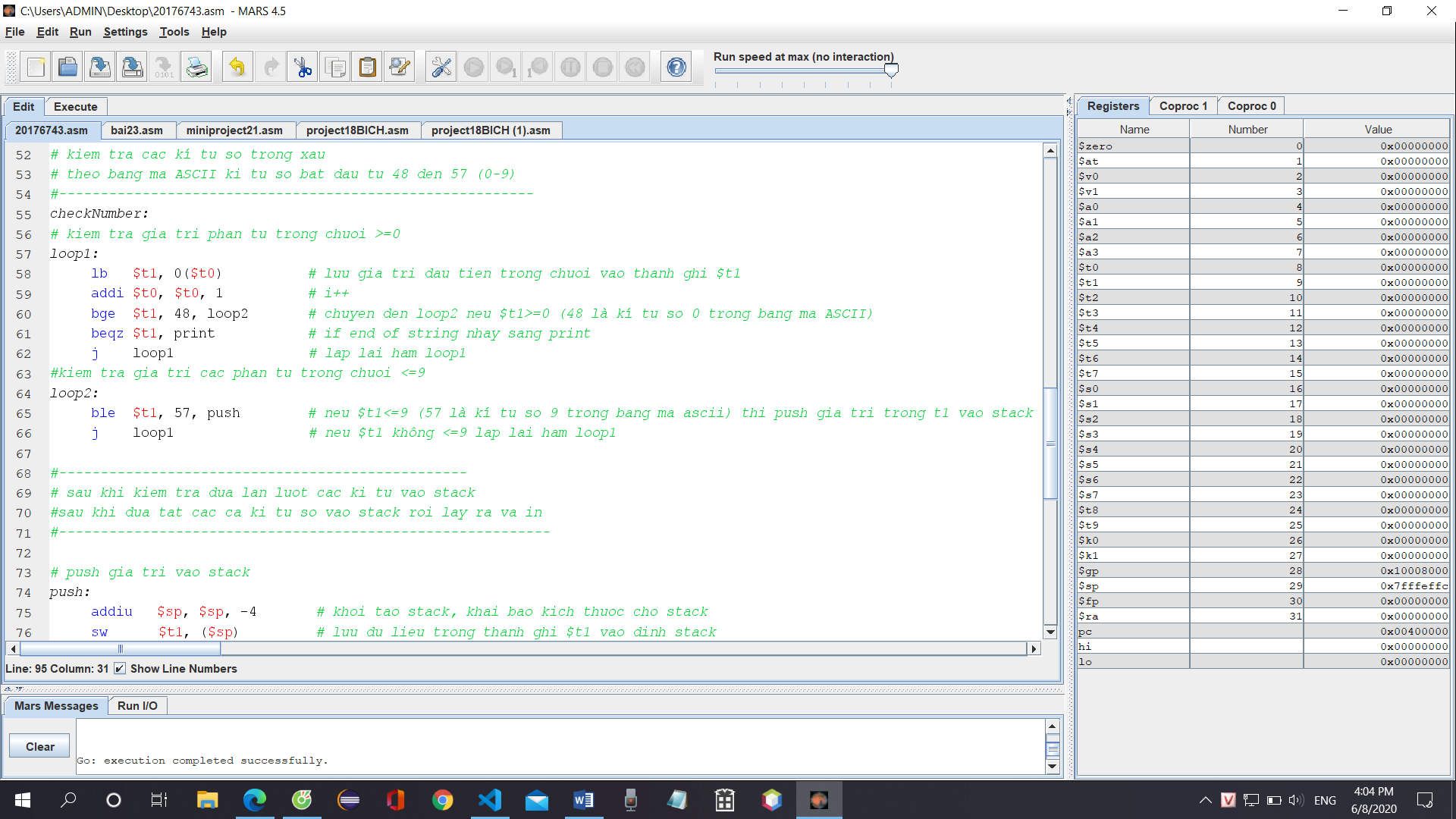


+ Hàm loop1: kiểm tra điều kiện kí tự >=0 (48 là kí tự 0 trong ASCII).

* Thanh ghi $t1: chứa giá trị của kí tự thứ i của xâu.
* Thanh ghi $t0: chứa giá trị thứ i+1.

+ Hàm loop2: kiểm tra điều kiện kí tự <=9 (57 là kí tự 9 trong ASCII).

* Thanh ghi $t1: chứa giá trị của kí tự thứ i của xâu.



+ Hàm push: push kí tự số sau khi thỏa mãn 2 điều kiện trên vào stack.

* Thanh ghi $t1: chứa kí tự số trước khi push vào stack.
* Thanh ghi $sp: lưu giá trị kí tự số vào stack.

+ Hàm pop: pop lần lượt các kí tự số đã cho vào stack và in ra kết quả.

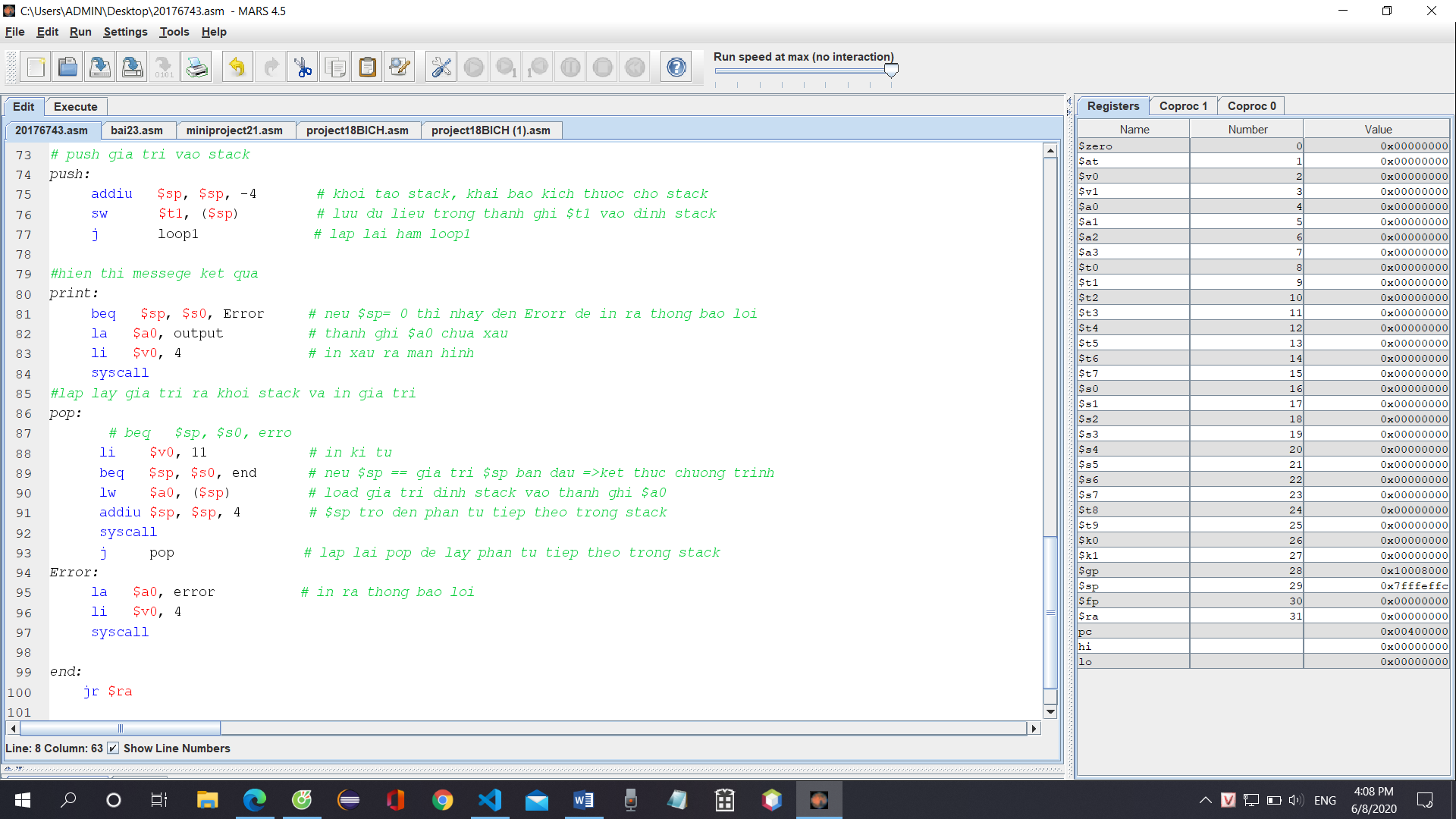
* Thanh ghi $s0: chứa giá trị thanh ghi ban đầu của stack.
* Thanh ghi $a0: chúa giá trị của kí tự sau khi đã lấy ra khỏi stack.

+ Hàm print:

* Hiển thị lời nhắn : “chuỗi số trong xâu ban đầu sau khi đã đảo ngược”.
* Nếu không có phần tử nào thỏa mãn điều kiện thì nhảy đến Erorr.

+ Hàm Erorr:

* Hiển thị lời nhắn: “chuoi ban vua nhap khong chua ki tu so".



ĐỀ TÀI 21 (BONUS PROJECTS)

Đề bài:

Let's define the digit degree of some positive integer as the number of times we need to replace this number with the sum of its digits until we get to a one digit number. Given an integer, find its digit degree.

Example

●For n = 5, the output should be digitDegree(n) =0;

●For n = 100, the output should be digitDegree(n) = 1. 1 + 0 + 0 = 1.

●For n = 91, the output should be digitDegree(n) = 2. 9 + 1 = 10-> 1 + 0 = 1.

Nội dung thực hiện:

1. Cách thực hiện.

* B1: Nhập số nguyên. Kiểm tra giá trị vừa nhập có là giá trị số sử dụng hàm CheckNumber.
* B2: Tính tổng các chữ số của số đó.Nếu tổng đó không phải là một số có 1 chữ số thì tiếp tục thực hiện tính tổng các chữ số của tổng đó. Lặp lại bao giờ ta được một tổng là số có một chữ số thì biến đếm cout chính là digit degree của số nguyên được nhập vào.

Đầu tiên xet xem số n nhập vào có phải là số hai chữ số không, nếu là 1 chữ số thì cho ra kết quả là 0,ngược lại thì tăng biến đếm và tính tổng các chữ số.

Để tính tổng các chữ số của một số ta sử dụng lệnh “div” kết quả được lưu vào hai thanh ghi “hi “và thanh ghi “lo” để tách các chữ số và thực hiện lệnh add để tính tổng.

Lấy tổng đó thay thế cho giá trị nhập vào.

* + - * B3: in ra kết quả.

#----------------------MÃ C--------------------------------------------

int n;

# printf("Nhap n:");

# scanf("%d", &n);

# int count = 0;

# while(n>=10){

# sum = 0;

# count++;

# while(n!=0){

# sum +=n%10

# n=n/10

# }

# n=sum;

# }

# return count;

#---------------------------------------------------------------------------

1. Ý nghĩa của các thanh ghi và các hàm được sử dụng.

* Sử dụng một số nhãn result, re2, loinhan để hiển thị ra kết quả.

Bước 1:

\*Nhãn message được sử dụng để hiển thị lời nhắn “Nhap vao n:". Ta sử dụng hàm checkNumber để kiểm tra giá trị nhập vào có là giá trị số không.

- Các thanh ghi được sử dụng trong các nhãn:

+ $s6: Cờ kiểm soát lỗi: s6=0 thì không có lỗi, s6=1 có lỗi và đưa ra thông báo nhập lại.

+ $t9: vị trí i của kí tự trong xâu.

+$s4: giá trị được chuyển từ xâu sang số

+ $t2: chứa Address string[i]

+$t3: chứa giá trị string[i]

+ $s2: gán giá trị ‘0’ cho s2

+ $s5: gán giá trị ‘9’ cho s5

+$t4: string[i] >= '0' ? $t4 = 1 : 0

+$t5: string[i] <= '9' ? $t5 = 1 : 0

+$t6: '0' <= string[i] <= '9' ? $t1 = 1 : 0

+$t7: lấy giá trị số của kí tự

+$ra: trở về chương trình chính

* Thanh ghi $s0 lưu trữ giá trị số nhập vào.

Bước 2:

* Hàm “digitDegree” có các vòng lặp while1,while2 ,nextwhile1 được thực hiện

+Thanh ghi $s1 để lưu giá trị biến đếm count.

+ Thanh ghi $t8 để lưu giá trị nhập vào phục vụ cho việc in ra kết quả.

+ Thanh ghi $s7 để lưu giá trị sum( tổng các chữ số của số nguyên n).

+ Thanh ghi $t0 để lưu giá trị của thanh ghi “hi” sau mỗi lần chia (gia tri dư của phép chia a % b )

+ Nhãn while1 để tính count.

+ Nhãn while để tính tổng các chữ sô của giá trị nhập vào.

+Nhãn nextWhile1 để thay thế giá trị nhập vào bằng tổng các chữ số của nó.

Bước 3:

\*Sử dụng nhãn printfResult

=> Ta in ra được kết quả là DegitDegree(n)= giá trị tính được. Ví dụ DegieDegree(94)=2.