Shape, square

Description automatically generated

**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

-----□□□□-----

Logo

Description automatically generated

**BÁO CÁO THỰC HÀNH**

**KIẾN TRÚC MÁY TÍNH**

|  |  |
| --- | --- |
| **GIẢNG VIÊN HƯỚNG DẪN :** | **Ths. LÊ BÁ VUI** |
| **SINH VIÊN :** | **NGUYỄN HOÀNG ANH MSSV: 20194474** |
|  | **TRẦN NAM DƯƠNG MSSV: 20194534** |

*Hà Nội, Ngày 20 Tháng 07 Năm 2022*

# Mục Lục

**Nội dung Trang**

**Bài 9** 2

Đề bài. 2

Phân tích yêu cầu 2

Cách làm 2

Mã nguồn 3

Kết quả hiển thị 9

**Bài 7** 10

Đề bài. 10

Phân tích yêu cầu 10

Cách làm 10

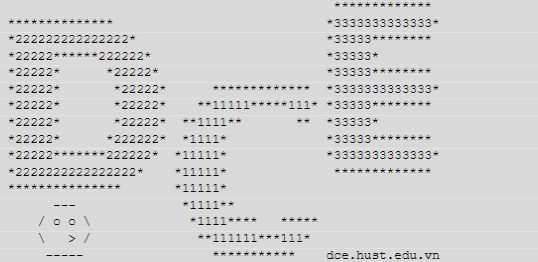
Mã nguồn 12

Kết quả hiển thị 21

# Bài 9: Vẽ hình bằng kí tự ASCII

## Sinh viên thực hiện: Nguyễn Hoàng Anh - 20194474

1. **Đề bài**: Cho hình ảnh đã được chuyển thành các kí tự ASCII như hình vẽ. Đây là hình của chữ DCE có viền \* và màu là các con số



- Hãy hiển thị hình ảnh trên lên giao diện console (hoặc giao diện Display trong công cụ giả lập Keyboard and Display MMIO Simulator)

- Hãy sửa ảnh để các chữ cái DCE chỉ còn lại viền, không còn màu số ở giữa, và hiển thị

- Hãy sửa ảnh để hoán đổi vị trí của các chữ, thành ECD, và hiển thị. Để đơn giản, các hoạ tiết đính kèm cũng được phép di chuyển theo.

- Hãy nhập từ bàn phím kí tự màu cho chữ D, C, E, rồi hiển thị ảnh trên với màu mới.

Chú ý: ngoài vùng nhớ lớn chứa ảnh được chứa sẵn trong code, không được tạo thêm vùng nhớ mới để chứa ảnh hiệu chỉnh .

## Phân tích yêu cầu

## Thiết kế chương trình với 1 menu chức năng bao gồm:

## + In ra màn hình hình ảnh 3 chữ DCE như trên đề bài

## + Chỉnh sửa ảnh, bỏ hết kí tự số, chỉ để lại viền chữ

## + In ra màn hình các chữ trong ảnh theo thứ tự ngược lại (ECD)

## + Nhập vào màu cho các chữ và hiển thị hình ảnh với màu mới

## Cách làm

Các bước làm trong bài bao gồm:

+ Khai báo các string chứa các đoạn kí tự để tạo thành hình ảnh các chữ D, C và E, đồng thời khai báo các string hay giá trị cần thiết phục vụ cho quá trình làm bài.

+ In menu các chức năng ra màn hình và cho phép chọn chức năng để thực hiện   
 + Đối với chức năng in ra 3 chữ: tạo vòng lặp chạy từ String1 đến String16 và in từng String ra màn hình.

+ Đối với chức năng in ra chữ rỗng: tương tự chức năng 1 là tạo vòng lặp chạy từ String1 đến String16, đổng thời tạo vòng lặp duyệt từng kí tự trong String, nếu kí tự được duyệt có mã ascii >47 và <58 (tương ứng với 0-> 9 trong bảng mã ascii) thì thay bằng dấu cách và in ra màn hình, còn không phải điều kiện trên thì sẽ in trực tiếp kí tự ra màn hình.

+ Đối với chức năng in ngược thứ tự 3 chữ: bằng cách chia từng string ra thành 3 đoạn tương ứng với 3 chữ, sau đó in theo thứ tự đoạn thứ 3 cho đến đoạn thứ nhất với thứ tự các đoạn là thành phần tạo thành các chữ tương ứng là D, C và E. Sau khi kết thúc vòng lặp duyệt và in 16 strings như 2 chức năng trên, sẽ cho ra thứ tự các chữ tương ứng trên màn hình là E C và D, đúng với yêu cầu đề bài.

+ Đối với chức năng đổi màu chữ: bước đầu là hỏi người dùng nhập vào màu (hay các số từ 1 -> 9) cho từng chữ tương ứng.

+ Sử dụng vòng lặp để lặp từ string1 đến string16 tương tự như các chức năng trên, đồng thời sử dụng vòng lặp qua từng kí tự trong 1 string, nếu là kí tự số 1->9 thì đổi màu (số) tương ứng đã khai báo từ đầu với màu vừa nhập vào. Khi kết thúc vòng lặp, kết quả in ra màn hình sẽ là màu được thay đổi đúng với yêu cầu đề bài

## Mã nguồn

.data

String1: .asciiz " \*\*\*\*\*\*\*\*\*\*\*\*\* \n"

String2: .asciiz "\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*3333333333333\* \n"

String3: .asciiz "\*222222222222222\* \*33333\*\*\*\*\*\*\*\* \n"

String4: .asciiz "\*22222\*\*\*\*\*\*22222\* \*33333\* \n"

String5: .asciiz "\*22222\* \*22222\* \*33333\*\*\*\*\*\*\*\* \n"

String6: .asciiz "\*22222\* \*22222\* \*\*\*\*\*\*\*\*\*\*\*\*\* \*3333333333333\* \n"

String7: .asciiz "\*22222\* \*22222\* \*\*11111\*\*\*\*\*111\* \*33333\*\*\*\*\*\*\*\* \n"

String8: .asciiz "\*22222\* \*22222\* \*\*1111\*\* \*\* \*33333\* \n"

String9: .asciiz "\*22222\* \*222222\* \*1111\* \*33333\*\*\*\*\*\*\*\* \n"

String10: .asciiz "\*22222\*\*\*\*\*\*\*22222\* \*11111\* \*3333333333333\* \n"

String11: .asciiz "\*2222222222222222\* \*11111\* \*\*\*\*\*\*\*\*\*\*\*\*\* \n"

String12: .asciiz "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*11111\* \n"

String13: .asciiz " ^^^ \*1111\*\* \n"

String14: .asciiz " ( o o ) \*1111\*\*\*\* \*\*\*\*\* \n"

String15: .asciiz " ( >) \*\*111111\*\*\*111\* \n"

String16: .asciiz " \*\*\*\*\*\*\*\*\*\*\* dce.hust.edu.vn \n"

Message0: .asciiz "------------MENU-----------\n"

ChucNang1: .asciiz"1. In ra cac chu\n"

ChucNang2: .asciiz"2. In ra cac chu rong\n"

ChucNang3: .asciiz"3. Dao nguoc vi tri cac chu\n"

ChucNang4: .asciiz"4. Doi mau cho chu\n"

Thoat: .asciiz"5. Exit\n"

Nhap: .asciiz"Nhap gia tri: "

ChuD: .asciiz"Nhap màu cho chu D(0->9): "

ChuC: .asciiz"Nhap màu cho chu C(0->9): "

ChuE: .asciiz"Nhap màu cho chu E(0->9): "

.text

####################

li $t5 50 #t5 mau chu hien tai cua chu D

li $t6 49 #t6 mau chu hien tai cua chu C

li $t7 51 #t7 mau chu hien tai cua chu E

main:

la $a0, Message0 # nhap menu

li $v0, 4

syscall

la $a0, ChucNang1

li $v0, 4

syscall

la $a0, ChucNang2

li $v0, 4

syscall

la $a0, ChucNang3

li $v0, 4

syscall

la $a0, ChucNang4

li $v0, 4

syscall

la $a0, Thoat

li $v0, 4

syscall

la $a0, Nhap

li $v0, 4

syscall

li $v0, 5

syscall

Case1menu:

addi $v1 $0 1

bne $v0 $v1 Case2menu

j Menu1

Case2menu:

addi $v1 $0 2

bne $v0 $v1 Case3menu

j Menu2

Case3menu:

addi $v1 $0 3

bne $v0 $v1 Case4menu

j Menu3

Case4menu:

addi $v1 $0 4

bne $v0 $v1 Case5menu

j Menu4

Case5menu:

addi $v1 $0 5

bne $v0 $v1 defaultmenu

j Exit

defaultmenu:

j main

####### Menu 1: in ra man hinh #########

Menu1:

addi $t0, $0, 0 #bien dem =0

addi $t1, $0, 16

la $a0,String1

Loop: beq $t1, $t0, main

li $v0, 4

syscall

addi $a0, $a0, 68 #string1 -> string2 -> ... -> string16

addi $t0, $t0, 1

j Loop

###### Menu 2: bo het so o giua chi giu lai vien ######

Menu2: addi $s0, $0, 0 #bien dem tung hàng =0

addi $s1, $0, 16

la $s2,String1 # $s2 la dia chi cua string1

Lap: beq $s1, $s0, main

addi $t0, $0, 0 # $t0 la bien dem tung kí tu cua 1 hàng =0

addi $t1, $0, 68 # $t1 max 1 hàng là 68 kí tu

In1hang:

beq $t1, $t0, End

lb $t2, 0($s2) # $t2 luu gia tri cua tung phan tu trong string1

# li $a1 47 #so -1 tuong duong vs gia tri 47

# li $a2 57 #so 9 tuong duong vs gia tri 57

bgt $t2, 47, Lonhon0 #neu > 0 thi nhay den Lonhon0

j Tmp

Lonhon0: bgt $t2, 57, Tmp #neu > 9 nua thi van ko doi

addi $t2 $0 0x20 # thay doi $t2 thanh dau cach

j Tmp

Tmp: li $v0, 11 # in tung ki tu

addi $a0 $t2 0

syscall

addi $s2 $s2 1 #sang chu tiep theo

addi $t0, $t0, 1# bien dem chu

j In1hang

End: addi $s0 $s0 1 # tang bien dem hàng lên 1

j Lap

###### Menu 3: doi vi tri chu ######

Menu3: addi $s0, $0, 0 #bien dem tung hàng =0

addi $s1, $0, 16

la $s2,String1 #$s2 luu dia chi cua string1

Lap2: beq $s1, $s0, main

#tao thanh 3 string nho

sb $0 21($s2)

sb $0 43($s2)

sb $0 65($s2)

#doi vi tri

li $v0, 4

la $a0 44($s2) #in chu E

syscall

li $v0, 4

la $a0 22($s2) # in chu C

syscall

li $v0, 4

la $a0 0($s2) # in chu D

syscall

li $v0, 4

la $a0 66($s2)

syscall

# ghep lai thanh string ban dau

addi $t1 $0 0x20

sb $t1 21($s2)

sb $t1 43($s2)

sb $t1 65($s2)

addi $s0 $s0 1

addi $s2 $s2 68

j Lap2

###### Menu 4: doi mau cho chu #######

Menu4:

NhapmauD: li $v0, 4

la $a0, ChuD

syscall

li $v0, 5 # lay mau cua ki tu D

syscall

blt $v0,0, NhapmauD

bgt $v0,9, NhapmauD

addi $s3 $v0 48 #$s3 luu mau cua chu D

NhapmauC: li $v0, 4

la $a0, ChuC

syscall

li $v0, 5 # lay mau cua ki tu C

syscall

blt $v0, 0, NhapmauC

bgt $v0, 9, NhapmauC

addi $s4 $v0 48 #$s4 luu mau cua chu C

NhapmauE: li $v0, 4

la $a0, ChuE

syscall

li $v0, 5 # lay mau cua ki tu E

syscall

blt $v0, 0, NhapmauE

bgt $v0, 9, NhapmauE

addi $s5 $v0 48 #$s5 luu mau cua chu E

addi $s0, $0, 0 #bien dem tung hàng = 0

addi $s1, $0, 16

la $s2,String1 # $s2 la dia chi cua string1

li $a1 48 #gia tri cua so 0

li $a2 57 #gia tri cua so 9

# li $t3 21

# li $t4 43

Lapdoimau: beq $s1, $s0, updatemau

addi $t0, $0, 0 # $t0 la bien dem tung kí tu cua 1 hàng =0

addi $t1, $0, 68 # $t1 max 1 hàng là 68 kí tu

In1hangdoimau:

beq $t1, $t0, Enddoimau

lb $t2, 0($s2) # $t2 luu gia tri cua tung phan tu trong string1

CheckD: bgt $t0, 21, CheckC #kiem tra het chu D chua

beq $t2, $t5, fixD

j Tmpdoimau

CheckC: bgt $t0, 43, CheckE #kiem tra het chu C chua

beq $t2, $t6, fixC

j Tmpdoimau

CheckE: beq $t2, $t7, fixE

j Tmpdoimau

fixD: sb $s3 0($s2)

j Tmpdoimau

fixC: sb $s4 0($s2)

j Tmpdoimau

fixE: sb $s5 0($s2)

j Tmpdoimau

Tmpdoimau: addi $s2 $s2 1 #sang chu tiep theo

addi $t0, $t0, 1# bien dem chu

j In1hangdoimau

Enddoimau: li $v0, 4

addi $a0 $s2 -68

syscall

addi $s0 $s0 1 # tang bien dem hàng lên 1

j Lapdoimau

updatemau: move $t5 $s3

move $t6 $s4

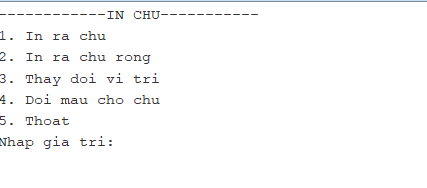
move $t7 $s5

j main

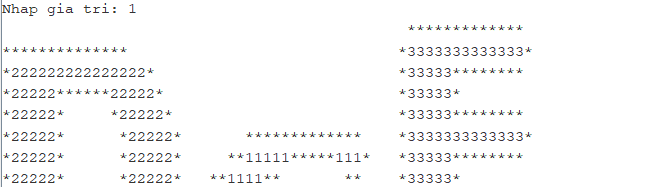
Exit:

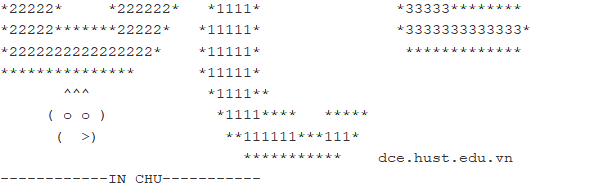
## Hình ảnh kết quả mô phỏng

*Menu các chức năng của chương trình*

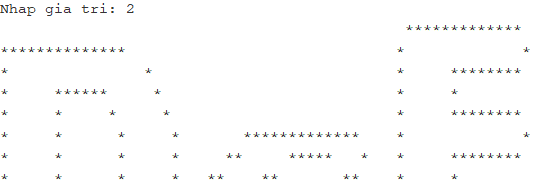
****

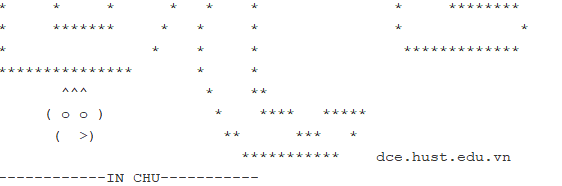
*Demo chức năng 1*



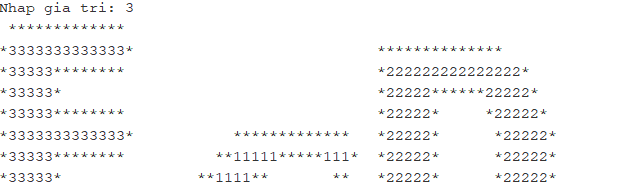


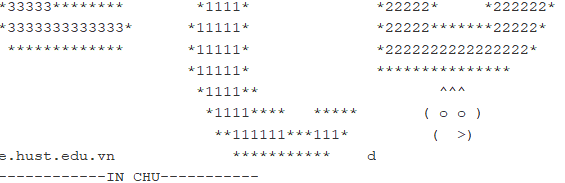
*Demo chức năng 2*

**

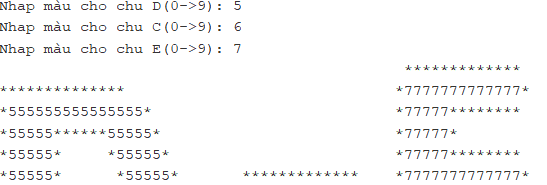
**

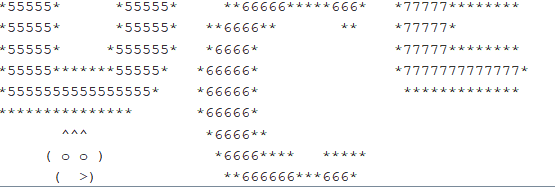
*Demo chức năng 3*

**

**

*Demo chức năng 4*

**

**

# Bài 7: chương trình kiểm tra cú pháp lệnh MIPS

## Sinh viên thực hiện: Trần Nam Dương – 20194534

1. **Đề bài** Trình biên dịch của bộ xử lý MIPS sẽ tiến hành kiểm tra cú pháp các lệnh hợp ngữ trong mã nguồn, xem có phù hợp về cú pháp hay không, rồi mới tiến hành dịch các lệnh ra mã máy. Hãy viết một chương trình kiểm tra cú pháp của 1 lệnh hợp ngữ MIPS bất kì (không làm với giả lệnh) như sau:

-Nhập vào từ bàn phím một dòng lệnh hợp ngữ. Ví dụ beq s1,31,t4

-Kiểm tra xem mã opcode có đúng hay không? Trong ví dụ trên, opcode là beq là hợp lệ thì hiện thị thông báo “opcode: beq, hợp lệ”

-Kiểm tra xem tên các toán hạng phía sau có hợp lệ hay không? Trong ví dụ trên, toán hạng s1 là hợp lệ, 31 là không hợp lệ, t4 thì khỏi phải kiểm tra nữa vì toán hạng trước đã bị sai rồi.

Gợi ý: nên xây dựng một cấu trúc chứa khuôn dạng của từng lệnh với tên lệnh, kiểu của toán hạng 1, toán hạng 2, toán hạng 3.

1. **Phân tích yêu cầu :**

+ Nhập vào bàn phím 1 lệnh hợp ngữ MIPS bất kì (không phải giả lệnh), kết quả hiển thị trên màn hình cho biết các thành phần của lệnh có hợp lệ hay không, từ đó suy ra lệnh có đúng cú pháp hay không.

### Cách làm

**Mô tả thuật toán**

đọc từ đầu đến khi gặp ' ' thì dừng lại lưu lại chuỗi vừa đọc được và so sánh với opCode để biết nó là lệnh gì để xử lý tiếp

đọc tiếp sau dấu ' ' mới nãy, đọc tiếp đến khi gặp ',' thì dừng lại và lưu chuỗi vừa rồi là toán hạng thứ nhất

tiếp tục đọc đến khi ',' thì dừng lại và lưu chuỗi vừa rồi là toán hạng thứ hai

vì mips chỉ có 3 toán toán hạng nên toán hạng cuối đọc đến hết chuỗi và lưu lại.

TH: opCode là lệnh J thì chỉ cần 1 toán hạng và nó theo dạng label nên viết hàm kiểm tra toán hạng 1 đã lưu lại dạng chuỗi có thoả mãn là 1 label ko (label chỉ chứ kí tự, số và dấu '\_ ' )

tiếp tục nếu toán hạng 1 thỏa mãn thì kiểm tra xem có tồn tại toán hạng 2 không? nếu có báo lỗi vì lệnh J chỉ cần 1 toán hạng

TH: opCode là lệnh add thì cần đến 3 toán hạng và đều là thanh ghi nên lấy 3 chuỗi của 3 toán hạng đọc được kiểm tra xem có phải là 1 trong 32 thanh ghi hay không? nếu 1 trong 3 ko là thanh ghi thì báo lỗi nếu lệnh nhập vào ban đầu ko đủ 3 toán hạng cũng báo lỗi vì thiếu toán hạng

TH: beq thì cũng tương tự, bắt buộc có 3 toán hạng, 2 toán hạng đầu kiểm tra có phải thanh ghi hay không? nếu ko thì báo lỗi. Kiểm tra toán hạng thứ 3 xem có thỏa mãn label ko nếu ko cũng báo lỗi

**Mã nguồn:**

.data

Nhap: .asciiz "Nhap vao mot dong lenh hop ngu: "

continueMessage: .asciiz "Ban muon tiep tuc chuong trinh?(0.Yes/1.No)"

NF: .asciiz "Khong tim duoc khuon dang lenh nay!\n"

endMess: .asciiz "\nHoan thanh! Lenh vua nhap vao phu hop voi cu phap!\n"

hopLe1: .asciiz "Opcode: "

hopLe11: .asciiz "Toan hang: "

hopLe2: .asciiz "hop le.\n"

command: .space 100

opcode: .space 10

token: .space 20

number: .space 15

ident: .space 30

# quy luat cua library: opcode co do dai = 5 byte

# moi lenh co 3 toan hang va chi co 4 loai la: thanh ghi = 1, hang so nguyen =2, dinh danh = 3 hoac khong co = 0.

library: .asciiz "or\*\*\*1111;xor\*\*1111;lui\*\*1201;jr\*\*\*1001;jal\*\*3002;addi\*1121;add\*\*1111;sub\*\*1111;ori\*\*1121;and\*\*1111;beq\*\*1132;bne\*\*1132;j\*\*\*\*3002;nop\*\*0001;"

charGroup: .asciiz "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ\_"

tokenRegisters: .asciiz "$zero $at $v0 $v1 $a0 $a1 $a2 $a3 $t0 $t1 $t2 $t3 $t4 $t5 $t6 $t7 $s0 $s1 $s2 $s3 $s4 $s5 $s6 $s7 $t8 $t9 $k0 $k1 $gp $sp $fp $ra $0 $1 $2 $3 $4 $5 $7 $8 $9 $10 $11 $12 $13 $14 $15 $16 $17 $18 $19 $20 $21 $22 $21 $22 $23 $24 $25 $26 $27 $28 $29 $30 $31 "

.text

readData: # Doc lenh nhap vao tu ban phim

li $v0, 4

la $a0, Nhap

syscall

li $v0, 8

la $a0, command # chua dia chi cua lenh nhap vao

li $a1, 100

syscall

main:

li $t2, 0 # i

readOpcode:

la $a1, opcode # luu cac ki tu doc duoc vao opcode

add $t3, $a0, $t2 # dich bit

add $t4, $a1, $t2

lb $t1, 0($t3) # doc tung ki tu cua command

sb $t1, 0($t4)

beq $t1, 32, done # gap ki tu ' ' -> luu ki tu nay vao opcode de xu ly

beq $t1, 0, done # ket thuc chuoi command

addi $t2, $t2, 1

j readOpcode

#<--xu ly opcode-->

done:

li $t7,-10

la $a2, library

xuLyOpcode:

li $t1, 0 # i

li $t2, 0 # j

addi $t7,$t7,10 # buoc nhay = 10 de den vi tri opcode trong library

add $t1,$t1,$t7 # cong buoc nhay

compare:

add $t3, $a2, $t1 # t3 tro thanh con tro cua library

lb $s0, 0($t3)

beq $s0, 0, notFound # khong tim thay opcode nao trong library

beq $s0, 42, check # gap ki tu '\*' -> check xem opcode co giong nhau tiep ko?.

add $t4, $a1, $t2

lb $s1, 0($t4)

bne $s0,$s1,xuLyOpcode # so sanh 2 ki tu. dung thi so sanh tiep, sai thi nhay den phan tu chua khuon danh lenh tiep theo.

addi $t1,$t1,1 # i+=1

addi $t2,$t2,1 # j+=1

j compare

check:

add $t4, $a1, $t2

lb $s1, 0($t4)

bne $s1, 32, check2 # neu ki tu tiep theo khong phai ' ' => lenh khong hop le. chi co doan dau giong.

checkContinue:

add $t9,$t9,$t2 # t9 = luu vi tri de xu ly token trong command

li $v0, 4

la $a0, hopLe1 # opcode hop le

syscall

li $v0, 4

la $a0, opcode

syscall

li $v0, 4

la $a0, hopLe2

syscall

j readToanHang1

check2: # neu ki tu tiep theo khong phai '\n' => lenh khong hop le. chi co doan dau giong.

bne $s1, 10, notFound

j checkContinue

# <!--ket thuc xu ly opcode -->

#<--xu li toan hang-->

readToanHang1:

# xac dinh kieu toan hang trong library

# t7 dang chua vi tri khuon dang lenh trong library

li $t1, 0

addi $t7, $t7, 5 # chuyen den vi tri toan hang 1 trong library

add $t1, $a2, $t7 # a2 chua dia chi library

lb $s0, 0($t1)

addi $s0,$s0,-48 # chuyen tu char -> int

li $t8, 1 # thanh ghi = 1

beq $s0, $t8, checkTokenReg

li $t8, 2 # hang so nguyen = 2

beq $s0, $t8, checkHSN

li $t8, 3 # dinh danh = 3

beq $s0, $t8, checkIdent

li $t8, 0 # khong co toan hang = 0

beq $s0, $t8, checkNT

j end

#<--check Token Register-->

checkTokenReg:

la $a0, command

la $a1, token # luu ten thanh ghi vao token de so sanh

li $t1, 0

li $t2, -1

addi $t1, $t9, 0

readToken:

addi $t1, $t1, 1 # i

addi $t2, $t2, 1 # j

add $t3, $a0, $t1

add $t4, $a1, $t2

lb $s0, 0($t3)

add $t9, $zero, $t1 # vi tri toan hang tiep theo trong command

beq $s0, 44, readTokenDone # gap dau ','

beq $s0, 0, readTokenDone # gap ki tu ket thuc

sb $s0, 0($t4)

j readToken

readTokenDone:

sb $s0, 0($t4) # luu them ',' vao de compare

li $t1, -1 # i

li $t2, -1 # j

li $t4, 0

li $t5, 0

add $t2, $t2, $k1

la $a1, token

la $a2, tokenRegisters

j compareToken

compareToken:

addi $t1,$t1,1

addi $t2,$t2,1

add $t4, $a1, $t1

lb $s0, 0($t4)

beq $s0, 0, end

add $t5, $a2, $t2

lb $s1, 0($t5)

beq $s1, 0, notFound

beq $s1, 32, checkLengthToken

bne $s0,$s1, jump

j compareToken

checkLengthToken:

beq $s0, 44, compareE

beq $s0, 10, compareE

j compareNE

jump:

addi $k1,$k1,6

j readTokenDone

compareE:

la $a0, hopLe11 # opcode hop le

syscall

li $v0, 4

la $a0, token

syscall

li $v0, 4

la $a0, hopLe2

syscall

addi $v1, $v1, 1 # dem so toan hang da doc.

li $k1, 0 # reset buoc nhay

beq $v1, 1, readToanHang2

beq $v1, 2, readToanHang3

beq $v1, 3, end

j end

compareNE:

j notFound

#<!--ket thuc check Token Register-->

#<--check toan hang la hang so nguyen-->

checkHSN: # kiem tra co phai hang so nguyen hay ko

la $a0, command

la $a1, number # luu day chu so vao number de so sanh tung chu so co thuoc vao numberGroup hay khong.

li $t1, 0

li $t2, -1

addi $t1, $t9, 0

readNumber:

addi $t1, $t1, 1 # i

addi $t2, $t2, 1 # j

add $t3, $a0, $t1

add $t4, $a1, $t2

lb $s0, 0($t3)

add $t9, $zero, $t1 # vi tri toan hang tiep theo trong command

beq $s0, 44, readNumberDone # gap dau ','

beq $s0, 0, readNumberDone # gap ki tu ket thuc

sb $s0, 0($t4)

j readNumber

readNumberDone:

sb $s0, 0($t4) # luu them ',' vao de compare

li $t1, -1 # i

li $t4, 0

la $a1, number

j compareNumber

compareNumber:

addi $t1, $t1, 1

add $t4, $a1, $t1

lb $s0, 0($t4)

beq $s0, 0, end

beq $s0, 45, compareNumber # bo dau '-'

beq $s0, 10, compareNumE

beq $s0, 44, compareNumE

li $t2, 48

li $t3, 57

slt $t5, $s0, $t2

bne $t5, $zero, compareNumNE

slt $t5, $t3, $s0

bne $t5, $zero, compareNumNE

j compareNumber

compareNumE:

la $a0, hopLe11

syscall

li $v0, 4

la $a0, number

syscall

li $v0, 4

la $a0, hopLe2

syscall

addi $v1, $v1, 1 # dem so toan hang da doc.

li $k1, 0 # reset buoc nhay

beq $v1, 1, readToanHang2

beq $v1, 2, readToanHang3

beq $v1, 3, end

j end

compareNumNE:

j notFound

#<!--ket thuc check toan hang la hang so nguyen-->

#<--check Indent-->

checkIdent:

la $a0, command

la $a1, ident # luu ten thanh ghi vao indent de so sanh

li $t1, 0

li $t2, -1

addi $t1, $t9, 0

readIndent:

addi $t1, $t1, 1 # i

addi $t2, $t2, 1 # j

add $t3, $a0, $t1

add $t4, $a1, $t2

lb $s0, 0($t3)

add $t9, $zero, $t1 # vi tri toan hang tiep theo trong command

beq $s0, 44, readIdentDone # gap dau ','

beq $s0, 0, readIdentDone # gap ki tu ket thuc

sb $s0, 0($t4)

j readIndent

readIdentDone:

sb $s0, 0($t4) # luu them ',' vao de compare

loopj:

li $t1, -1 # i

li $t2, -1 # j

li $t4, 0

li $t5, 0

add $t1, $t1, $k1

la $a1, ident

la $a2, charGroup

j compareIdent

compareIdent:

addi $t1,$t1,1

add $t4, $a1, $t1

lb $s0, 0($t4)

beq $s0, 0, end

beq $s0, 10, compareIdentE

beq $s0, 44, compareIdentE

loop:

addi $t2,$t2,1

add $t5, $a2, $t2

lb $s1, 0($t5)

beq $s1, 0, compareIdentNE

beq $s0, $s1, jumpIdent # so sanh ki tu tiep theo trong ident

j loop # tiep tuc so sanh ki tu tiep theo trong charGroup

jumpIdent:

addi $k1,$k1,1

j loopj

compareIdentE:

la $a0, hopLe11 # opcode hop le

syscall

li $v0, 4

la $a0, ident

syscall

li $v0, 4

la $a0, hopLe2

syscall

addi $v1, $v1, 1 # dem so toan hang da doc.

li $k1, 0 # reset buoc nhay

beq $v1, 1, readToanHang2

beq $v1, 2, readToanHang3

beq $v1, 3, end

j end

compareIdentNE:

j notFound

#<!--ket thuc check Indent-->

#<--kiem tra khong co toan hang-->

checkNT:

la $a0, command

li $t1, 0

li $t2, 0

addi $t1, $t9, 0

add $t2, $a0, $t1

lb $s0, 0($t2)

addi $v1, $v1, 1 # dem so toan hang da doc.

li $k1, 0 # reset buoc nhay

beq $v1, 1, readToanHang2

beq $v1, 2, readToanHang3

beq $v1, 3, end

#<!--ket thuc kiem tra khong co toan hang-->

#<--check Token Register 2-->

readToanHang2:

# xac dinh kieu toan hang trong library

# t7 dang chua vi tri khuon dang lenh trong library

li $t1, 0

la $a2, library

addi $t7, $t7, 1 # chuyen den vi tri toan hang 2 trong library

add $t1, $a2, $t7 # a2 chua dia chi library

lb $s0, 0($t1)

addi $s0,$s0,-48 # chuyen tu char -> int

li $t8, 1 # thanh ghi = 1

beq $s0, $t8, checkTokenReg

li $t8, 2 # hang so nguyen = 2

beq $s0, $t8, checkHSN

li $t8, 3 # dinh danh = 3

beq $s0, $t8, checkIdent

li $t8, 0 # khong co toan hang = 0

beq $s0, $t8, checkNT

j end

#<!--ket thuc check Token Register 2-->

#<--check Token Register 3-->

readToanHang3:

# xac dinh kieu toan hang trong library

# t7 dang chua vi tri khuon dang lenh trong library

li $t1, 0

la $a2, library

addi $t7, $t7, 1 # chuyen den vi tri toan hang 3 trong library

add $t1, $a2, $t7 # a2 chua dia chi library

lb $s0, 0($t1)

addi $s0,$s0,-48 # chuyen tu char -> int

li $t8, 1 # thanh ghi = 1

beq $s0, $t8, checkTokenReg

li $t8, 2 # hang so nguyen = 2

beq $s0, $t8, checkHSN

li $t8, 3 # dinh danh = 3

beq $s0, $t8, checkIdent

li $t8, 0 # khong co toan hang = 0

beq $s0, $t8, checkNT

j end

#<!--ket thuc check Token Register 3-->

#<--ket thuc xu li toan hang-->

continue: # lap lai chuong trinh.

li $v0, 4

la $a0, continueMessage

syscall

li $v0, 5

syscall

add $t0, $v0, $zero

beq $t0, $zero, resetAll

j TheEnd

resetAll:

li $v0, 0

li $v1, 0

li $a0, 0

li $a1, 0

li $a2, 0

li $a3, 0

li $t0, 0

li $t1, 0

li $t2, 0

li $t3, 0

li $t4, 0

li $t5, 0

li $t6, 0

li $t7, 0

li $t8, 0

li $t9, 0

li $s0, 0

li $s1, 0

li $s2, 0

li $s3, 0

li $s4, 0

li $s5, 0

li $s6, 0

li $s7, 0

li $k0, 0

li $k1, 0

j readData

notFound:

li $v0, 4

la $a0, NF

syscall

j continue

end:

li $v0, 4

la $a0, endMess

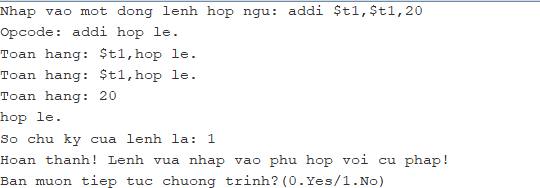
syscall

j continue

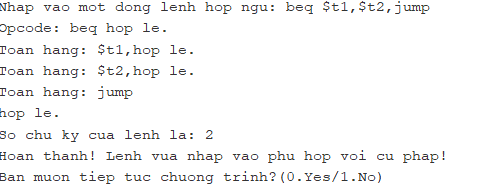
TheEnd:

1. **Hình ảnh kết quả mô phỏng:**

*Kiểm tra lệnh addi:*

****

*Kiểm tra lệnh beq:*

**