**Mục Lục**

**Nội dung Trang**

**Bài 5**............................................................................................... 2

Đề bài.................................................................................... 2

Thuật toán............................................................................ 2

Mã nguồn.............................................................................. 4

Kết quả hiển thị..................................................................... 18

**Bài 9**............................................................................................... 19

Đề bài.................................................................................... 19

Trình bày............................................................................... 20

Thuật toán............................................................................. 20

Mã nguồn.............................................................................. 21

Kết quả hiển thị..................................................................... 29

**Bài 5: Biểu thức trung tố hậu tố**

**Sinh viên thực hiện: Nguyễn Địch Long – 20162525**

**Đề bài: Viết chương trình tính giá trị biểu thức bất kỳ bằng phương pháp duyệt biểu thức hậu tố.**

**Các yêu cầu cụ thể:**

1. **Nhập vào biểu thức trung tố, ví dụ: 9 + 2 + 8 \*6**
2. **In ra biểu thức ở dạng hậu tố, ví dụ: 9 2 + 8 6 \* +**
3. **Tính ra giá trị của biểu thức vừa nhập**

**Các hằng số là số nguyên, trong phạm vi từ 0->99**

**Toán tử bao gồm phép cộng, trừ, nhân, chia lấy thương.**

**Thuật toán:**

1. **Đổi biểu thức trung tố sang hậu tố:**

Để đổi biểu thức trung tố sang hậu tố, ta sẽ dùng ngăn xếp và xâu

**B1:** Đưa một biểu thức trung tố vào xâu kí tự và đặt tên là str

**B2:** Tạo ra một xâu mới để lưu biểu thức hậu tố, đặt tên là str2

**B3:**  Sắp xếp lại xâu:

* Nếu kí tự là số thì lưu vào str2
* Nếu kí tự là toán tử, nếu ngăn xếp trống thì đẩy vào ngăn xếp
* Nếu toán tử đang xét có bậc cao hơn toán tử ở đỉnh ngăn xếp thì đẩy toán tử vào ngăn xếp
* Nếu toán tử đang xét có bậc bằng toán tử ở đỉnh ngăn xếp thì lấy toán tử đỉnh ngăn xếp ra, xếp vào str2 và đẩy toán tử đang xét vào ngăn xếp
* Nếu toán tử đang xét có bậc nhỏ hơn toán tử ở đỉnh ngăn xếp thì lấy toán tử đang xét xếp vào str2

**B4:** Thực hiện bước 3 cho đến khi kết thúc biểu thức và tất cả các toán tử toán hạng được xếp vào str2, khi đó ta có biểu thức hậu tố.

1. **Tính giá trị biểu thức hậu tố:**

**B1:** Quét toàn bộ biểu thức hậu tố từ trái sang phải

**B2:** Tạo một ngăn xếp mới

**B3:** Nếu phần tử được quét là toán hạng thì đưa vào ngăn xếp

**B4:** Nếu phần tử được quét là toán tử thì lấy 2 toán hạng trong ngăn xếp ra, sau đó tính toán giá trị của chúng dựa vào toán tử này, sau đó đẩy lại vào ngăn xếp

**B5:** Thực hiện bước 3 và 4 cho đến khi kết thúc biểu thức và trong ngăn xếp còn 1 giá trị duy nhất. Đó chính là giá trị biểu thức hậu tố

**Mã nguồn:**

.data

infix:.space256

postfix:.space256

stack:.space256

prompt: .asciiz "Enter String contain infix expression¥n(note) Input expression has number must be integer and positive number:"

newLine:.asciiz"¥n"

prompt\_postfix:.asciiz"Postfixis:"

prompt\_result:.asciiz"Resultis:"

prompt\_infix:.asciiz"Infixis:"

#getinfix

.text

li$v0,54

la$a0,prompt la$a1,infix

la$a2,256

syscall

la$a0,prompt\_infix

li$v0,4

syscall

la$a0,infix

li$v0,4

syscall

#converttopostfix

li$s6,-1#counter

li$s7,-1#Scounter

li$t7,-1#Pcounter

while:

la$s1,infix #buffer=$s1

la$t5,postfix#postfix=$t5

la$t6,stack#stack=$t6

li$s2,'+'

li$s3,'-'

li$s4,'\*'

li$s5,'/'

addi$s6,$s6,1 #counter++

#getbuffer[counter]

add$s1,$s1,$s6

lb$t1,0($s1) #t1=valueofbuffer[counter]

beq$t1,$s2,operator#'+'

nop

beq$t1,$s3,operator#'-'

nop

beq$t1,$s4,operator#'\*'

nop

beq$t1,$s5,operator#'/'

nop

beq$t1,10,n\_operator#'¥n'

nop

beq$t1,32,n\_operator#''

nop

beq$t1,$zero,endWhile

nop

#pushnumbertopostfix a

ddi$t7,$t7,1 a

dd$t5,$t5,$t7

sb$t1,0($t5)

lb$a0,1($s1)

jalcheck\_number

beq$v0,1,n\_operator

nop

add\_space:

add$t1,$zero,32 sb$t1,1($t5)

addi$t7,$t7,1

jn\_operator

nop

operator:

#addtostack...

beq$s7,-1,pushToStack

nop

add$t6,$t6,$s7

lb$t2,0($t6)#t2=valueofstack[counter]

#checkt1precedence

beq$t1,$s2,t1to1

nop

beq$t1,$s3,t1to1

nop

li$t3,2

jcheck\_t2

nop

t1to1:

li$t3,1

#checkt2precedence

check\_t2:

beq$t2,$s2,t2to1

nop

beq$t2,$s3,t2to1

nop

li$t4,2

jcompare\_precedence

nop

t2to1:

li$t4,1

compare\_precedence:

beq$t3,$t4,equal\_precedence

nop

slt$s1,$t3,$t4

beqz $s1,t3\_large\_t4

nop

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

#t3<t4

#popt2fromstack andt2==>postfix

#getnewtopstackdoagain

sb$zero,0($t6)

addi$s7,$s7,-1 #scounter++

addi$t6,$t6,-1

la$t5,postfix#postfix=$t5 addi$t7,$t7,1

add$t5,$t5,$t7

sb$t2,0($t5)

#addi$s7,$s7,-1#scounter=scounter-1

joperator

nop

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

t3\_large\_t4:

#pusht1tostack

jpushToStack

nop

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

equal\_precedence:

#popt2 fromstack andt2==>postfix

#pushtostack

sb$zero,0($t6)

addi$s7,$s7,-1 #scounter++

addi$t6,$t6,-1

la$t5,postfix#postfix=$t5

addi$t7,$t7,1#pcounter++

add$t5,$t5,$t7

sb$t2,0($t5)

jpushToStack

nop

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

pushToStack:

la$t6,stack#stack=$t6

addi$s7,$s7,1 #scounter++

add$t6,$t6,$s7

sb$t1,0($t6)

n\_operator:

jwhile

nop

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

endWhile:

addi$s1,$zero,32

add$t7,$t7,1

add$t5,$t5,$t7

la$t6,stack

add$t6,$t6,$s7

popallstack:

lb$t2,0($t6)#t2=valueofstack[counter]

beq$t2,0,endPostfix

sb$zero,0($t6)

addi$s7,$s7,-2

add$t6,$t6,$s7

sb$t2,0($t5)

add$t5,$t5,1

jpopallstack

nop

endPostfix:

############################################################ ###################ENDPOSTFIX

#printpostfix

la$a0,prompt\_postfix

li$v0,4

syscall

la$a0,postfix

li$v0,4

syscall

la$a0,newLine

li$v0,4

syscall

############################################################ ###################Caculate

li$s3,0#counter

la$s2,stack#stack=$s2

#postfixtostack

while\_p\_s:

la$s1,postfix#postfix=$s1

add$s1,$s1,$s3

lb$t1,0($s1)

#ifnull

beqz$t1end\_while\_p\_s

nop

add$a0,$zero,$t1

jalcheck\_number

nop

beqz$v0,is\_operator

nop

jaladd\_number\_to\_stack

nop

jcontinue

nop

is\_operator:

jalpop

nop

add$a1,$zero,$v0#b

jalpop

nop

add$a0,$zero,$v0#a

add$a2,$zero,$t1#op

jalcaculate

continue:

add$s3,$s3,1#counter++

jwhile\_p\_s

nop

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

#Procedurecaculate #@briefcaculatethenumber("aopb")

#@param[int]a0:(int)a

#@param[int]a1:(int)b

#@param[int]a2:operator(op)ascharacter

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

caculate:

sw$ra,0($sp)

li$v0,0

beq$t1,'\*',cal\_case\_mul

nop

beq$t1,'/',cal\_case\_div nop

beq$t1,'+',cal\_case\_plus

nop

beq$t1,'-',cal\_case\_sub

cal\_case\_mul:

mul$v0,$a0,$a1

jcal\_push

cal\_case\_div:

div$a0,$a1

mflo$v0

jcal\_push

cal\_case\_plus:

add$v0,$a0,$a1

jcal\_push

cal\_case\_sub:

sub$v0,$a0,$a1

jcal\_push

cal\_push:

add$a0,$v0,$zero

jalpush

nop

lw$ra,0($sp)

jr$ra

nop

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

#Procedureadd\_number\_to\_stack#@briefgetthenumberandaddnumbertostackat$s2

#@param[in]s3:counterforpostfixstring

#@param[in]s1:postfixstring #@param[in]t1:currentvalue

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

add\_number\_to\_stack:

#save$ra

sw$ra,0($sp)

li$v0,0

while\_ants:

beq$t1,'0',ants\_case\_0

nop

beq$t1,'1',ants\_case\_1

nop

beq$t1,'2',ants\_case\_2

nop

beq$t1,'3',ants\_case\_3

nop

beq$t1,'4',ants\_case\_4

nop

beq$t1,'5',ants\_case\_5

nop

beq$t1,'6',ants\_case\_6

nop

beq$t1,'7',ants\_case\_7

nop

beq$t1,'8',ants\_case\_8

nop

beq$t1,'9',ants\_case\_9

nop

ants\_case\_0:

jants\_end\_sw\_c

ants\_case\_1:

addi$v0,$v0,1

jants\_end\_sw\_c

nop

ants\_case\_2:

addi$v0,$v0,2

jants\_end\_sw\_c

nop

ants\_case\_3:

addi$v0,$v0,3

jants\_end\_sw\_c

nop

ants\_case\_4:

addi$v0,$v0,4

jants\_end\_sw\_c

nop

ants\_case\_5:

addi$v0,$v0,5

jants\_end\_sw\_c

nop

ants\_case\_6:

addi$v0,$v0,6

jants\_end\_sw\_c

nop

ants\_case\_7:

addi$v0,$v0,7

jants\_end\_sw\_c

nop

ants\_case\_8:

addi$v0,$v0,8

jants\_end\_sw\_c

nop

ants\_case\_9:

addi$v0,$v0,9

jants\_end\_sw\_c

nop

ants\_end\_sw\_c:

add$s3,$s3,1#counter++

la$s1,postfix#postfix=$s1

add$s1,$s1,$s3

lb$t1,0($s1)

beq$t1,$zero,end\_while\_ants

beq$t1,'',end\_while\_ants

mul$v0,$v0,10

jwhile\_ants

end\_while\_ants:

add$a0,$zero,$v0

jalpush

#get$ra lw$ra,0($sp)

jr$ra

nop

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

#Procedurecheck\_number

#@briefcheckcharacterisnumberornot

#@param[int]a0:charactertocheck

#@param[out]v0:1=true;0=false

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

check\_number:

li$t8,'0'

li$t9,'9'

beq$t8,$a0,check\_number\_true

beq$t9,$a0,check\_number\_true

slt$v0,$t8,$a0

beqz$v0,check\_number\_false

slt$v0,$a0,$t9

beqz$v0,check\_number\_false

check\_number\_true:

li$v0,1

jr$ra

nop

check\_number\_false:

li$v0,0

jr$ra

nop

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

#Procedurepop

#@briefpopfromstackat$s2

#@param[out]v0:valuetopopped

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

pop:

lw$v0,-4($s2)

sw$zero,-4($s2)

add$s2,$s2,-4

jr$ra

nop

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

#Procedurepush

#@briefpushtostackat$s2

#@param[in]a0:valuetopush

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

push:

sw$a0,0($s2)

add$s2,$s2,4

jr$ra

nop

end\_while\_p\_s:

#addnulltoendofstack

#printpostfix la$a0,prompt\_result

li$v0,4

syscall

jal pop

add$a0,$zero,$v0

li$v0,1

syscall

la$a0,newLine

li$v0,4

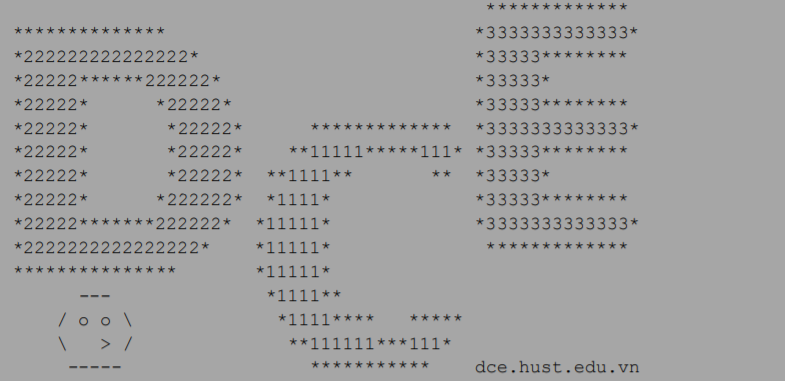
syscall

**Kết quả hiển thị :**

**Bài 9: Vẽ hình bằng kí tự ASCII**

**Sinh viên thực hiện: Trần Văn Thông – 20167386**

**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, cà hiển thị
* Hãy sửa ảnh để hoán đổi vị trí của các chữ, thành ECD, và hiển thị. Để đơn giản, các họa 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 hình ảnh trên 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.

**Trình bày:**

* Tạo menu để dễ dàng thao tác hơn

**Menu:**

**----------------** Ve hinh bang ki tu ASCII**----------------**

1. Hien thi hinh anh
2. Hien thi hinh anh ko co mau
3. Doi vi tri cua D va E
4. Doi mau
5. Thoat

Select option:

**Thuật toán:**

* Option 1: Sử dụng vòng lặp để in ra string của từng dòng, đến khi hết 16 dòng chuyển về Menu.
* Option2: Duyệt từng kí tự theo từng dòng, nếu không phải số (>0 hoặc >9) thì in ra kí tự đó. Nếu là chữ số thì thay chữ số đó bằng space (dấu cách ) để xóa màu.
* Option3: Chia hình ảnh ra thành 4 phần:
* Phần 1 (Chữ D): Từ cột 0 đến cột 23 🡨 Gán char ở cột 23 = NULL
* Phần 2 (Chữ C): Từ cột 24 đến cột 44 🡨 Gán char ở cột 44 = NULL
* Phần 3 (Chữ E) : Từ cột 45 đến cột 61 🡨 Gán char ở cột 61 = NULL
* Phần 4: Các cột còn lại (space và \n)

Sau đó in theo từng hàng lần lượt các kí tự từ cột 45 -> cột 61 (Chữ E),

cột 24 -> cột 44 (Chữ C), cột 0 -> cột 23 (chữ D). Rồi trả lại các cột 23, 44, 61 về space.

* Option4: Sau khi bam Option 4 sẽ hiện ra màn hình lần lượt:

4.1. Doi mau cho D (0->9):

4.2. Doi mau cho C (0->9):

4.3. Doi mau cho E (0->9):

Để người dùng chọn màu để đổi cho D, C, E.

Lưu các kí tự người dùng nhập vào rồi duyệt từng kí tự theo từng dòng.

Xét theo từng chữ cái, với chữ D (0->23) nếu gặp màu cũ (kí tự số cũ) thì đổi sang màu mới (kí tự số người dùng nhập). Tương tự với C và E.

* Option 5: Kết thúc chương trình

**Mã nguồn:**

.data

String0: .space 5000

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

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

String3: .asciiz " \*2222222222222222\* \*33333\*\*\*\*\*\*\*\* \n"

String4: .asciiz " \*22222\*\*\*\*\*\*\*222222\* \*33333\* \n"

String5: .asciiz " \*22222\* \*222222\* \*33333\*\*\*\*\*\*\*\* \n"

String6: .asciiz " \*22222\* \*222222\* \*\*\*\*\*\*\*\*\*\*\*\*\* \*3333333333333\* \n"

String7: .asciiz " \*22222\* \*222222\* \*\*11111\*\*\*\*\*111\* \*33333\*\*\*\*\*\*\*\* \n"

String8: .asciiz " \*22222\* \*222222\* \*1111\*\* \*\*\* \*33333\* \n"

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

String10: .asciiz " \*22222\*\*\*\*\*\*\*222222\* \*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"

Stack: .space 1000

Message0: .asciiz "------------Ve hinh bang ki tu ASCII------------\n"

Option1: .asciiz"1. Hien thi hinh anh\n"

Option2: .asciiz"2. Hien thi hinh anh ko co mau\n"

Option3: .asciiz"3. Doi vi tri cua D va E\n"

Option4: .asciiz"4. Doi mau\n"

Exit\_Menu: .asciiz"5. Thoat\n"

Select: .asciiz"Select option: "

StringD: .asciiz"4.1. Doi mau cho D (0->9): "

StringC: .asciiz"4.2. Doi mau cho C (0->9): "

StringE: .asciiz"4.3. Doi mau cho E (0->9): "

.text

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

li $t5 50 #t5 mau ban dau cua D

li $t6 49 #t6 mau ban dau cua C

li $t7 51 #t7 mau ban dau cua E

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

main:

################################################# bat dau in man hinh menu

la $a0, Message0 # In Menu

li $v0, 4 # Goi print

syscall

la $a0, Option1 # Option 1 In string

li $v0, 4

syscall

la $a0, Option2 # Option 2 In string khong co mau

li $v0, 4

syscall

la $a0, Option3 # Option 3 Doi vi tri cua D va E

li $v0, 4

syscall

la $a0, Option4 # Option 4 Doi mau

li $v0, 4

syscall

la $a0, Exit\_Menu # Option 5 Thoat menu

li $v0, 4

syscall

la $a0, Select # Chon option

li $v0, 4

syscall

########################################## ket thuc in menu

li $v0, 5 #Doc so duoc nhap tu ban phim roi luu vao $v0

syscall

########################################## bat dau xu ly cac case

Case1menu:

addi $v1 $0 1 # if select option 1

bne $v0 $v1 Case2menu

j Menu1

Case2menu:

addi $v1 $0 2 # if select option 2

bne $v0 $v1 Case3menu

j Menu2

Case3menu:

addi $v1 $0 3 # if select option 3

bne $v0 $v1 Case4menu

j Menu3

Case4menu:

addi $v1 $0 4 # if select option 4

bne $v0 $v1 Case5menu

j Menu4

Case5menu:

addi $v1 $0 5 # if select option 5

bne $v0 $v1 defaultmenu

j Exit

defaultmenu:

j main # Neu chon so khong hop le thi nhay ve Menu

############# In ####################

Menu1:

addi $s0, $0, 0 # init count variable i = 0

addi $s1, $0, 16 # 16 la tong so hang

la $s2 , String1 # load string cua hang dau tien vao $s2

Loop\_Menu1:

beq $s1, $s0, main # if print hoan thanh tro ve menu

la $a0, ($s2) # else in ra string cua hang dang luu o $s2

li $v0, 4

syscall

addi $s2, $s2, 64 # nhay sang dia chi cua string cua hang tiep theo vi 1 hang co 64 ki tu

addi $s0, $s0, 1 # next sang hang tiep theo

j Loop\_Menu1

############ In khong co mau (xoa mau) ################

Menu2: addi $s0, $0, 0 # variable i dem cac hang

addi $s1, $0, 16 # so hang

la $s2, String1 # Gan $s2 cho xau cua hang 1

Loop\_Menu2: #for i=0;i<16

beq $s1, $s0, main

addi $t0, $0, 0 # variable j dem cot

addi $t1, $0, 64 # so cot

menu2\_In1hang:

beq $t1, $t0, menu2\_Endline # neu j==64 thi ket thuc 1 hang

lb $t2, 0($s2) # load 1 ki tu

bgt $t2, '0', menu2\_compare\_with\_9 # if ki tu duoc luu o $t2 > '0' thi nhay vao ham menu2\_Compare\_With\_9 de kiem tra tiep voi 9

j menu2\_print\_char #neu char < '0' thi in ra char

menu2\_compare\_with\_9:

bgt $t2, '9', menu2\_print\_char # if char > '9' thi in ra char

addi $t2, $0, 0x20 # neu char la number thi gan = space

j menu2\_print\_char # jump to menu 2

menu2\_print\_char:

li $v0, 11 # in ki tu

addi $a0,$t2,0 # print $t2

syscall

addi $s2 $s2 1 # ki tu tiep theo

addi $t0, $t0, 1# Tang bien dem cot len 1

#beq $t2, '\*', menu2\_storeStar

j menu2\_In1hang

menu2\_Endline:

addi $s0 $s0 1 # tang hang len 1

j Loop\_Menu2

################# Doi vi tri D va E ############

Menu3: addi $s0, $0, 0 # bien dem cac hang

addi $s1, $0, 16 # so hang

la $s2, String1 # Gan $s2 cho string cua hang 1

Loop2: beq $s1, $s0, main # neu doc het rows thi quay ve main

# Phan thanh 4 vung

sb $0 23($s2) #gan ki tu o string[23] = null

sb $0 44($s2) #gan ki tu o string[44] = null

sb $0 61($s2) #gan ki tu o string[61] = null

# Doi vi tri

li $v0, 4

la $a0 45($s2) # in cac ki tu cua E ra dau tien bang cach doc string tu String[45] den ki tu null tai String[61]

syscall

li $v0, 4

la $a0 24($s2) # in cac ki tu cua C ra bang cach doc string tu String[24] den ki tu null tai String[44]

syscall

li $v0, 4

la $a0 0($s2) # in cac ki tu cua D ra bang cach doc string tu String[0] den ki tu null tai String[23]

syscall

li $v0, 4 # in cac ki tu space va xuong dong` o cuoi line tu String [61] den ki tu null tai String [63]

la $a0 62($s2)

syscall

# Chuoi co so

addi $t1 $0 0x20

sb $t1 23($s2) # tra lai gia tri space cho cac gia tri bi gan la null luc truoc

sb $t1 44($s2)

sb $t1 61($s2)

addi $s0 $s0 1 # tang so hang len 1

addi $s2 $s2 64 # doc dia chi cua String cua hang tiep theo

j Loop2

############ doi mau cho chu ################

Menu4:

Input\_D\_Color: li $v0, 4

la $a0, StringD # hoi yeu cau nguoi dung nhap vao mau cho D

syscall

li $v0, 5 # nhan vao gia tri tu nguoi dung read integer

syscall

blt $v0, 0, Input\_D\_Color # neu gtri <0 thi yeu cau nguoi dung nhap lai

bgt $v0, 9, Input\_D\_Color # neu gtri >9 thi yeu cau nguoi dung nhap lai

addi $s3, $v0, 48 # $s3 store D's color

Input\_C\_Color: li $v0, 4

la $a0, StringC # hoi yeu cau nguoi dung nhap vao mau cho C

syscall

li $v0, 5 # nhan vao gia tri tu nguoi dung read integer

syscall

blt $v0, 0, Input\_C\_Color # neu gtri <0 thi yeu cau nguoi dung nhap lai

bgt $v0, 9, Input\_C\_Color # neu gtri >9 thi yeu cau nguoi dung nhap lai

addi $s4, $v0, 48 # $s3 store C's color

Input\_E\_Color: li $v0, 4

la $a0, StringE # hoi yeu cau nguoi dung nhap vao mau cho E

syscall

li $v0, 5 # nhan vao gia tri tu nguoi dung read integer

syscall

blt $v0, 0, Input\_C\_Color # neu gtri <0 thi yeu cau nguoi dung nhap lai

bgt $v0, 9, Input\_C\_Color # neu gtri >9 thi yeu cau nguoi dung nhap lai

addi $s5, $v0, 48 # $s3 store E's color

addi $s0, $0, 0 # count rows

addi $s1, $0, 16 # number rows

la $s2,String1 # Doc vao String cua row dau tien

Loop\_Change\_Color:

beq $s1, $s0, Update\_ColorCode # neu doc het row thi luu lai gia tri cua new color roi quay ve main

addi $t0, $0, 0 # doc tung ki tu

addi $t1, $0, 64 # so ki tu max cua 1 row

Loop\_Row\_Change\_Color:

beq $t1, $t0, Enddoimau\_row # neu doc het row thi in row ra va next sang row tiep theo

lb $t2, 0($s2) # load char

CheckD: # char cua D tu String[0] den String[23]

bgt $t0, 23, CheckC # neu ki tu o column >23 thi check C

beq $t2, $t5, fixD # if current char == D base color thi doi mau D

j Next\_Char

CheckC: # char cua C tu String[24] den String[44].

bgt $t0, 44, CheckE # neu ki tu o column >44 thi check E

beq $t2, $t6, fixC # if current char == C base color thi doi mau C

j Next\_Char

CheckE: # char cua E tu String[45] den String[63]

beq $t2, $t7, fixE # if current char == E base color thi doi mau E

j Next\_Char

fixD: sb $s3 0($s2) # gan new color vao char hien tai

j Next\_Char

fixC: sb $s4 0($s2) # gan new color vao char hien tai

j Next\_Char

fixE: sb $s5 0($s2) # gan new color vao char hien tai

j Next\_Char

Next\_Char: addi $s2 $s2 1 # to next Char

addi $t0, $t0, 1 # increase count variable column

j Loop\_Row\_Change\_Color

Enddoimau\_row:

li $v0, 4

addi $a0 $s2 -64 # print string coulumn

syscall

addi $s0 $s0 1 # increase rows

j Loop\_Change\_Color

Update\_ColorCode: # neu doc het row thi luu lai gia tri cua new color roi quay ve main

move $t5 $s3

move $t6 $s4

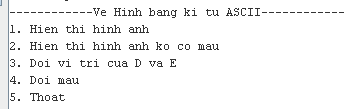
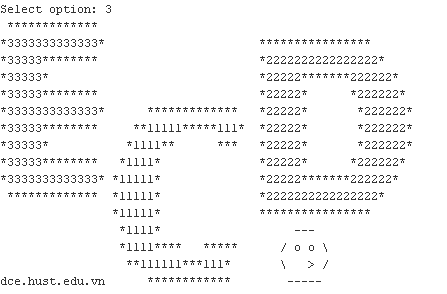
move $t7 $s5

j main

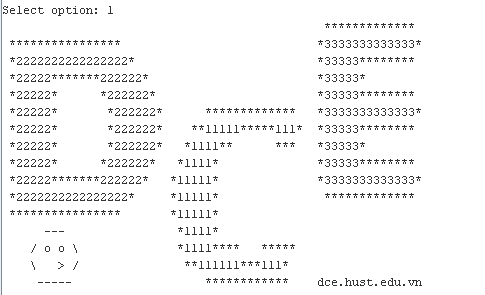
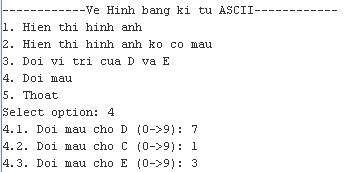
Exit:

**Kết quả hiển thị:**

Menu Kết quả của Option 3

Kết quả của Option 1 Giao diện Menu sau khi chọn Option4

Kết quả của Option 2 Kết quả của Option 4

