TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
ĐẠI HỌC BÁCH KHOA HÀ NỘI

BÁO CÁO CUỐI KỲ

Thực Hành Kiến Trúc Máy Tính

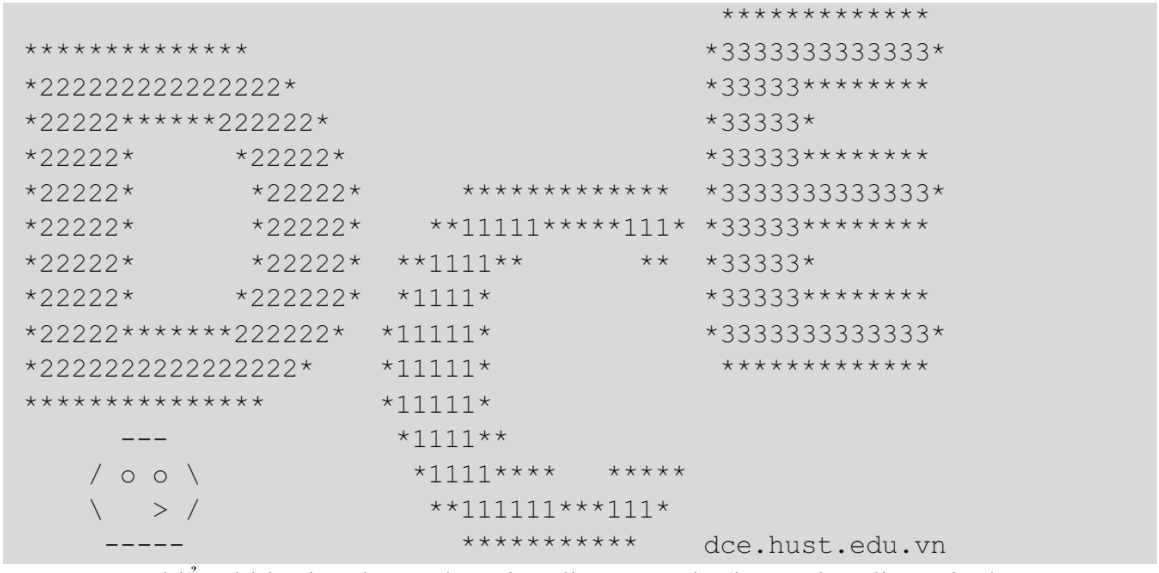
Giáo viên hướng dẫn: ThS.Lê Bá Vui
Sinh viên thực hiện: Phan Đức Duy – 20225831 – Bài 9
Nguyễn Đức Đại Dương – 20225616 – Bài 8
Nhóm: 17
Lớp: 147789

Mục Lục

A. Phan Đức Duy – Bài 9

I. 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.

II. Mã Nguồn

```
.data
        m:      .space 5000
        menu:   .asciiz "\nmenu: \n1. in hinh anh goc\n2. in hinh anh
khong co mau\n3. in hinh anh dao nguoc chu\n4. doi mau chu\n5. exit\n"
        m7:     .asciiz "mau cua D(0-9): "
        m8:     .asciiz "mau cua C(0-9): "
        m9:     .asciiz "mau cua E(0-9): "
        m10:    .asciiz "yeu cau khong hop le vui long chon tu 1 den
5\n"
        m11:    .asciiz "so khong hop le vui long chon tu 0 den 9\n"
.text
continue:
main:
#------------------hien thi menu--------------------#
        li      $v0, 4
        la      $a0, menu
        syscall
        # menu:
#------------------nhap yeu cau---------------------#
        li      $v0, 5
        syscall
        add     $s4, $zero, $v0
        # luu yeu cau nguoi dung vao s4
#----------------kiem tra dieu kien-----------------#
        bge     $s4, 6, error
        ble     $s4, 0, error
        # neu nguoi dung nhap so khong hop le (lon hon 5 hoac nho hon 1)
thi nhay den error
        bne     $s4, 5, setup
        # neu yeu cau nguoi dung la 1,2,3,4 thi nhay den setup
        # neu yeu cau la 5 thi thuc hien exit
exit:
        li      $v0, 10
        syscall
        # thoat chuong trinh
#-------------thiet lap bien, tham so---------------#
setup:
        la      $s0, m # con tro
        la      $s1, 0 # bo dem
        la      $s2, 0 # so lan in ky tu
        li      $t0, 42 # *
        li      $t1, 50 # mau cua D
        li      $t2, 49 # mau cua C
        li      $t3, 51 # mau cua E
        li      $t4, 32 # space
        li      $t5, 10 # /n
```

```
        bne     $s4, 4, create_image
        # neu yeu cau nguoi dung la 1,2,3 thi thuc hien tao hinh

#-------------------doi mau chu---------------------#
        li      $v0, 4
        la      $a0, m7
        syscall
        # mau cua D(0-9):
        li      $v0, 5
        syscall
        addi    $t1, $v0, 48
        # luu mau cua D vao t1
        bge     $t1, 58, error2
        ble     $t1, 47, error2
        # kiem tra ky tu khong hop le


        li      $v0, 4
        la      $a0, m8
        syscall
        # mau cua C(0-9):
        li      $v0, 5
        syscall
        addi    $t2, $v0, 48
        # luu mau cua C vao t1
        bge     $t2, 58, error2
        ble     $t2, 47, error2
        # kiem tra ky tu khong hop le


        li      $v0, 4
        la      $a0, m9
        syscall
        # mau cua E(0-9):
        li      $v0, 5
        syscall
        addi    $t3, $v0, 48
        # luu mau cua E vao t1
        bge     $t3, 58, error2
        ble     $t3, 47, error2
        # kiem tra ky tu khong hop le
        j create_image

#------------------ham tao ky tu--------------------#
# cac ham ben duoi nhan tham so (s2) la so ky tu duoc in ra

create_border: # tao vien "*"
        sb      $t0, 0($s0)                     # luu ky tu * vao bo nho
        addi    $s0, $s0, 1                     # di chuyen con tro den o
nho tiep theo
        addi    $s1, $s1, 1                     # bo dem + 1
        bne     $s1, $s2, create_border
        # neu bo dem != s2(so ky tu dc in ra) thi tiep tuc in
        li      $s1, 0                          # thiet lap lai s1 ve 0
        jr      $ra

create_color_D: # tao mau cho D
```

```
                # cach hoat dong giong ham tren
        beq     $s4, 2, create_space
        # neu nguoi dung chon "2. in hinh anh khong co mau" thi tao
khoang trong
        sb      $t1, 0($s0)
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        bne     $s1, $s2, create_color_D
        li      $s1, 0
        jr      $ra
create_color_C: # tao mau cho D
                # cach hoat dong giong ham tren
        beq     $s4, 2, create_space
        # neu nguoi dung chon "2. in hinh anh khong co mau" thi tao
khoang trong
        sb      $t2, 0($s0)
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        bne     $s1, $s2, create_color_C
        li      $s1, 0
        jr      $ra
create_color_E: # tao mau cho D
                # cach hoat dong giong ham tren
        beq     $s4, 2, create_space
        # neu nguoi dung chon "2. in hinh anh khong co mau" thi tao
khoang trong
        sb      $t3, 0($s0)
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        bne     $s1, $s2, create_color_E
        li      $s1, 0
        jr      $ra
create_space:   # tao khoang trong
                # cach hoat dong giong ham tren
        sb      $t4, 0($s0)
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        bne     $s1, $s2, create_space
        li      $s1, 0
        jr      $ra
#------------------ham tao hinh---------------------#
create_image:
        # chieu rong cua D la 22
        # chieu rong cua C la 20
        # chieu rong cua E la 16

        li      $s2, 22                         # so ky tu duoc in la 22
        jal     create_space                    # nhay den ham in ky tu
tuong ung

        li      $s2, 20
        jal     create_space

        li      $s2, 1
        jal     create_space
        li      $s2, 13
        jal     create_border
```

```
li      $s2, 2
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 1

li      $s2, 14
jal     create_border
li      $s2, 8
jal     create_space

li      $s2, 20
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 13
jal     create_color_E
li      $s2, 1
jal     create_border
li      $s2, 1
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 2

li      $s2, 1
jal     create_border
li      $s2, 15
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_space

li      $s2, 20
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_E
li      $s2, 8
jal     create_border
li      $s2, 2
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 3

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 6
jal     create_border
li      $s2, 6
```

```
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 3
jal     create_space

li      $s2, 20
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_E
li      $s2, 1
jal     create_border
li      $s2, 9
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 4

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 6
jal     create_space
li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 2
jal     create_space

li      $s2, 20
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_E
li      $s2, 8
jal     create_border
li      $s2, 2
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 5

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
```

```
li      $s2, 1
jal     create_border
li      $s2, 7
jal     create_space
li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 1
jal     create_space

li      $s2, 5
jal     create_space
li      $s2, 13
jal     create_border
li      $s2, 2
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 13
jal     create_color_E
li      $s2, 1
jal     create_border
li      $s2, 1
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 6

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 7
jal     create_space
li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 1
jal     create_space

li      $s2, 3
jal     create_space
li      $s2, 2
jal     create_border
li      $s2, 5
jal     create_color_C
li      $s2, 5
jal     create_border
```

```
li      $s2, 3
jal     create_color_C
li      $s2, 1
jal     create_border
li      $s2, 1
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_E
li      $s2, 8
jal     create_border
li      $s2, 2
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 7

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 7
jal     create_space
li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 1
jal     create_space

li      $s2, 1
jal     create_space
li      $s2, 2
jal     create_border
li      $s2, 4
jal     create_color_C
li      $s2, 2
jal     create_border
li      $s2, 7
jal     create_space
li      $s2, 2
jal     create_border
li      $s2, 2
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_E
li      $s2, 1
jal     create_border
```

```
li      $s2, 9
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 8

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 6
jal     create_space
li      $s2, 1
jal     create_border
li      $s2, 6
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 1
jal     create_space

li      $s2, 1
jal     create_space
li      $s2, 1
jal     create_border
li      $s2, 4
jal     create_color_C
li      $s2, 1
jal     create_border
li      $s2, 13
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_E
li      $s2, 8
jal     create_border
li      $s2, 2
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 9

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_D
li      $s2, 7
jal     create_border
li      $s2, 6
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 2
```

```
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_C
li      $s2, 1
jal     create_border
li      $s2, 13
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 13
jal     create_color_E
li      $s2, 1
jal     create_border
li      $s2, 1
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 10

li      $s2, 1
jal     create_border
li      $s2, 16
jal     create_color_D
li      $s2, 1
jal     create_border
li      $s2, 4
jal     create_space

li      $s2, 1
jal     create_border
li      $s2, 5
jal     create_color_C
li      $s2, 1
jal     create_border
li      $s2, 13
jal     create_space

li      $s2, 1
jal     create_space
li      $s2, 13
jal     create_border
li      $s2, 2
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 11

li      $s2, 15
jal     create_border
li      $s2, 7
jal     create_space

li      $s2, 1
```

```
jal     create_border
li      $s2, 5
jal     create_color_C
li      $s2, 1
jal     create_border
li      $s2, 13
jal     create_space

li      $s2, 16
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 12

li      $s2, 6
jal     create_space
li      $s3, 45
sb      $s3, 0($s0)
addi    $s0, $s0, 1
sb      $s3, 0($s0)
addi    $s0, $s0, 1
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s2, 13
jal     create_space

li      $s2, 1
jal     create_space
li      $s2, 1
jal     create_border
li      $s2, 4
jal     create_color_C
li      $s2, 2
jal     create_border
li      $s2, 12
jal     create_space

li      $s2, 16
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 13

li      $s2, 4
jal     create_space
li      $s3, 47
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s2, 1
jal     create_space
li      $s3, 111
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s2, 1
jal     create_space
li      $s3, 111
sb      $s3, 0($s0)
```

```
addi    $s0, $s0, 1
li      $s2, 1
jal     create_space
li      $s3, 92
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s2, 11
jal     create_space

li      $s2, 2
jal     create_space
li      $s2, 1
jal     create_border
li      $s2, 4
jal     create_color_C
li      $s2, 4
jal     create_border
li      $s2, 3
jal     create_space
li      $s2, 5
jal     create_border
li      $s2, 1
jal     create_space

li      $s2, 16
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 14

li      $s2, 4
jal     create_space
li      $s3, 92
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s2, 3
jal     create_space
li      $s3, 62
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s2, 1
jal     create_space
li      $s3, 47
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s2, 1
jal     create_space
li      $s2, 10
jal     create_space

li      $s2, 3
jal     create_space
li      $s2, 2
jal     create_border
li      $s2, 6
jal     create_color_C
li      $s2, 3
```

```
jal     create_border
li      $s2, 3
jal     create_color_C
li      $s2, 1
jal     create_border
li      $s2, 2
jal     create_space

li      $s2, 16
jal     create_space
sb      $t5, 0($s0)
addi    $s0, $s0, 1
# dong 15

li      $s2, 5
jal     create_space
li      $s3, 45
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 45
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 45
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 45
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 45
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s2, 12
jal     create_space

li      $s2, 5
jal     create_space
li      $s2, 11
jal     create_border
li      $s2, 4
jal     create_space

li      $s3, 100
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 99
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 101
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 46
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 104
sb      $s3, 0($s0)
addi    $s0, $s0, 1
li      $s3, 117
```

```
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 115
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 116
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 46
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 101
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 100
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 117
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 46
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 118
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s3, 110
        sb      $s3, 0($s0)
        addi    $s0, $s0, 1
        li      $s2, 1
        jal     create_space
        # dong 16


        li      $s6, 16         # bien dem so dong
        la      $s0, m
        beq     $s4, 3, print_option_3
        # neu nguoi dung chon "3. in hinh anh dao nguoc chu"
#------------------in hinh 1, 2, 4---------------------#
print_option_1_2:
        li      $v0, 11
        lb      $a0, 0($s0)
        syscall
        # in ky tu
        beq     $a0, 0, end
        # gap ky tu NULL thi ket thuc
        addi    $s0, $s0, 1
        # con tro + 1
        j       print_option_1_2
#--------------------in hinh 3----------------------#
# ham nay thuc hien in hinh dao nguoc chu ECD
# chieu rong cua D la 22
# chieu rong cua C la 20
# chieu rong cua E la 16
print_option_3:
        addi    $s0, $s0, 42 # chuyen con tro den chu E
        li      $s1, 0       # thiet lap bien dem ve 0
```

```
loop:   # thuc hien in mot dong cua chu E
        li      $v0, 11
        lb      $a0, 0($s0)
        syscall
        addi    $s1, $s1, 1
        addi    $s0, $s0, 1
        bne     $s1, 16, loop   # in den khi nao du 16 ky tu
        addi    $s0, $s0, -36   # chuyen con tro den chu C
        li      $s1, 0          # thiet lap bien dem ve 0
loop2:  # thuc hien in mot dong cua chu C
        li      $v0, 11
        lb      $a0, 0($s0)
        syscall
        addi    $s1, $s1, 1
        addi    $s0, $s0, 1
        bne     $s1, 20, loop2  # in den khi nao du 20 ky tu
        addi    $s0, $s0, -42   # chuyen con tro den chu D
        li      $s1, 0          # thiet lap bien dem ve 0
loop3:  # thuc hien in mot dong cua chu D
        li      $v0, 11
        lb      $a0, 0($s0)
        syscall
        addi    $s1, $s1, 1
        addi    $s0, $s0, 1
        bne     $s1, 22, loop3  # in den khi nao du 22 ky tu
        addi    $s0, $s0, 79    # chuyen con tro den chu E cua dong tiep
theo
        li      $s1, 0          # thiet lap bien dem ve 0
        addi    $s6, $s6, -1    # so dong -1
        li      $v0, 11
        li      $a0, 10
        syscall
        # xuong dong
        bne     $s6, 0, loop
        j       end             # in du 16 dong thi ket thuc
#---------------------bao loi-----------------------#
error:
        li      $v0, 4
        la      $a0, m10
        syscall
        # bao loi "yeu cau khong hop le vui long chon tu 1 den 5"
        li      $v0, 4
        la      $a0, menu
        syscall
        # hien thi lai menu lua chon

        li      $v0, 5
        syscall
        add     $s4, $zero, $v0
        # doc lai yeu cau

        bge     $s4, 6, error
        ble     $s4, 0, error
        # tiep tuc kiem tra loi
        bne     $s4, 5, setup
        # neu chon 5 thi ket thuc
error2:
```

```
        li      $v0, 4
        la      $a0, m11
        syscall
        # bao loi "so khong hop le vui long chon tu 1 den 9"
        j       setup
        # chon lai mau
end:
        j       main
```

III. Kết Quả Mô Phỏng

1. Hiển thị menu

- Người dùng cần chọn 1 trong 5 lựa chọn trên để yêu cầu chương trình thực hiện

```
menu:
1. in hinh anh goc
2. in hinh anh khong co mau
3. in hinh anh dao nguoc chu
4. doi mau chu
5. exit
```

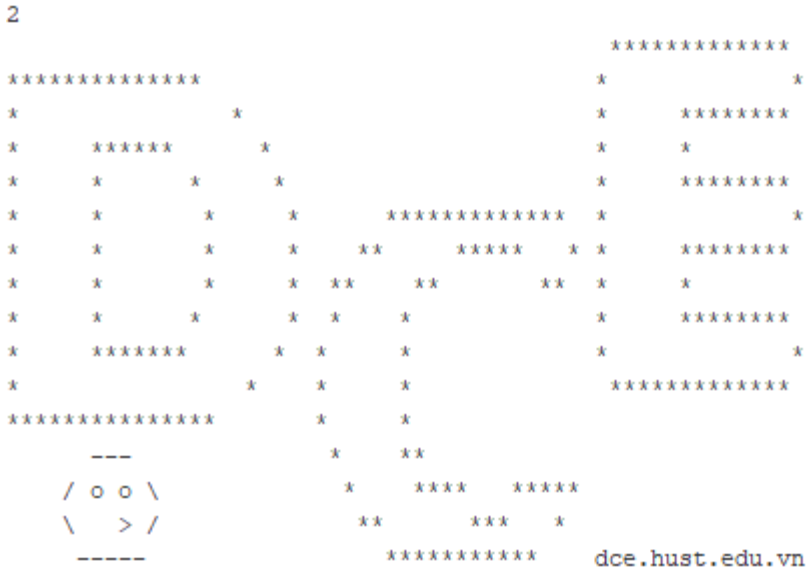
2. Lựa chọn 1

- khi chọn lựa chọn 1 tức là người dùng yêu cầu chương trình hiển thị lại hình ảnh gốc
- sau khi hiển thị hình ảnh lên màn hình chương trình sẽ tiếp tục đưa ra các lựa chọn cho người dùng

```
1
                                                 *************
**************                                  *3333333333333*
*222222222222222*                                *33333********
*22222******222222*                              *33333*
*22222*      *22222*                             *33333********
*22222*       *22222*      *************        *3333333333333*
*22222*       *22222*    **llllll*****lll*      *33333********
*22222*       *22222*  **llll**       **        *33333*
*22222*      *222222*  *llll*                    *33333********
*22222*******222222*  *llllll*                  *3333333333333*
*2222222222222222*    *llllll*                   *************
***************       *llllll*
      ---              *llll**
    / o o \             *llll****   *****
    \   > /              **lllllll***lll*
     -----                 **********     dce.hust.edu.vn
menu:
1. in hinh anh goc
2. in hinh anh khong co mau
3. in hinh anh dao nguoc chu
4. doi mau chu
5. exit
```

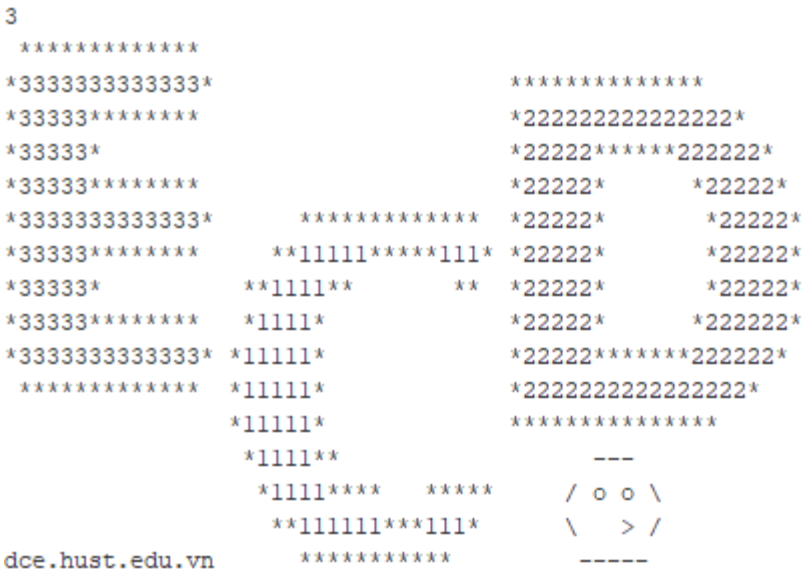
3. Lựa chọn 2

- khi đưa ra lựa chọn 2 tức là người dùng muốn hiển thị hình ảnh mà không chứa màu chữ (là những số bên trong của chữ)
- sau khi hiển thị hình ảnh lên màn hình chương trình sẽ tiếp tục đưa ra các lựa chọn cho người dùng



4. Lựa chọn 3

- Khi đưa ra lựa chọn 3 tức là người dùng muốn hiển thị hình ảnh là chữ ECD thay vì DCE, đồng thời các hình ảnh kèm theo chữ cũng đi theo chữ đó
- sau khi hiển thị hình ảnh lên màn hình chương trình sẽ tiếp tục đưa ra các lựa chọn cho người dùng



5. Lựa chọn 4

- khi đưa ra lựa chọn 4 tức là người dùng muốn đổi màu của chữ (màu là các số hiển thị bên trong chữ, ở đây là các số từ 0 đến 9)
- Chương trình lần lượt hiển thị yêu cầu người dùng nhập màu cho chữ D, C, E là các số từ 0 đến 9
- Người dùng không được nhập các số nằm ngoài các số được cho phép, nếu nhập sai chương trình sẽ báo lỗi và yêu cầu người dùng chọn màu lại từ đầu
- sau khi hiển thị hình ảnh lên màn hình chương trình sẽ tiếp tục đưa ra các lựa chọn cho người dùng

Đây là khi người dùng nhập đúng theo yêu cầu

```
4
mau cua D(0-9): 5
mau cua C(0-9): 2
mau cua E(0-9): 9
                                                 *************
**************                                  *9999999999999*
*555555555555555*                                *99999********
*55555******555555*                              *99999*
*55555*      *55555*                             *99999********
*55555*       *55555*      *************         *9999999999999*
*55555*       *55555*    **22222*****222*        *99999********
*55555*       *55555*  **2222**       **         *99999*
*55555*      *555555*  *2222*                     *99999********
*55555*******555555*  *22222*                     *9999999999999*
*5555555555555555*    *22222*                      *************
***************       *22222*
      ---              *2222**
    / o o \             *2222****   *****
    \   > /              **222222***222*
     -----                 ***********    dce.hust.edu.vn
menu:
1. in hinh anh goc
2. in hinh anh khong co mau
3. in hinh anh dao nguoc chu
4. doi mau chu
5. exit
```

Đây là khi người dùng nhập sau yêu cầu

```
4
mau cua D(0-9): 6
mau cua C(0-9): 8
mau cua E(0-9): 11
so khong hop le vui long chon tu 0 den 9
mau cua D(0-9):
```

6. Lựa chọn 5

- khi đưa ra lựa chọn 5 tức là người dùng yêu cầu dừng chương trình lại

```
menu:
1. in hinh anh goc
2. in hinh anh khong co mau
3. in hinh anh dao nguoc chu
4. doi mau chu
5. exit
5

-- program is finished running --
```

7. người dùng nhập số không có trong menu

- khi người dùng đưa ra lựa chọn không có trong menu, chương trình sẽ báo lỗi và yêu cầu người dùng nhập lại

```
menu:
1. in hinh anh goc
2. in hinh anh khong co mau
3. in hinh anh dao nguoc chu
4. doi mau chu
5. exit
6
yeu cau khong hop le vui long chon tu 1 den 5
1. in hinh anh goc
2. in hinh anh khong co mau
3. in hinh anh dao nguoc chu
4. doi mau chu
5. exit
```

IV. Thuật Toán

1. Tạo hình gốc

- Đầu tiên chương trình yêu cầu người dùng đưa ra các lựa chọn có trong menu, giả sử người dùng chọn lựa chọn 1, khi đó chương trình sẽ thực hiện khởi tạo các biến và tham số sau đó sẽ nhảy đến hàm create_image
- Tại create_image chương trình sử dụng phương pháp là tạo từng ký tự một và lưu chúng vào vùng nhớ

```
create_image:
        # chieu rong cua D la 22
        # chieu rong cua C la 20
        # chieu rong cua E la 16

        li      $s2, 22                         # so ky tu duoc in la 22
        jal     create_space                    # nhay den ham in ky tu tuong ung

        li      $s2, 20
        jal     create_space

        li      $s2, 1
        jal     create_space
        li      $s2, 13
        jal     create_border
        li      $s2, 2
        jal     create_space
        sb      $t5, 0($s0)
        addi    $s0, $s0, 1
        # dong 1
```

- Chúng ta tạo từng ký tự theo trình tự từ trái sang phải và từ trên xuống dưới
- ở ví dụ trên ta đang tạo hàng 1 của hình, ta gán cho s2 giá trị là số ký tự được in ra sau đó nhảy đến hàm tạo tương ứng như create_space, create_color_D, create_color_C, create_color_E,create_border

```
create_space:   # tao khoang trong
                # cach hoat dong giong ham tren
        sb      $t4, 0($s0)
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        bne     $s1, $s2, create_space
        li      $s1, 0
        jr      $ra
```

- ở hàm này nó sẽ lưu từng ký tự tương ứng vào ô nhớ sau đó tịnh tiến s0(con trỏ của ô nhớ) và s1 (bộ đếm) cho đến khi s1 == s2 thì dừng lại và quay trở lại hàm create_image

2. Tạo hình không màu

- Nếu người chọn lựa chọn 2 quy trình thực hiện sẽ giống lựa chọn một, tuy nhiên mỗi khi chương trình nhảy đến hàm create_color_ tương ứng hàm đó sẽ kiểm tra s4 (là lựa chọn của người dùng) nếu s4 == 2 thì sẽ nhảy đến hàm create_space để thay thế

```
create_color_D: # tao mau cho D
                # cach hoat dong giong ham tren
        beq     $s4, 2, create_space
        # neu nguoi dung chon "2. in hinh anh khong co mau" thi tao khoang trong
        sb      $t1, 0($s0)
        addi    $s0, $s0, 1
        addi    $s1, $s1, 1
        bne     $s1, $s2, create_color_D
        li      $s1, 0
        jr      $ra
```

3. Tạo hình Đảo Ngược

- ở trong vùng nhớ mỗi dòng chữ D sẽ chiếm 22 ô, C chiếm 20 ô, E chiếm 16 ô và kế tiếp là ký tự xuống dòng “\n” và lặp lại như vậy đến khi hết hình.
- Như vậy để có thể in hình ECD thay vì DCE ta chỉ cần tại mỗi dòng bắt đầu tin từ ký tự E sau đó quay lại ký tự C và quay lại ký tự D sau đó lại xuống ký tự E ở dòng tiếp theo.

```
print_option_3:
        addi    $s0, $s0, 42 # chuyen con tro den chu E
        li      $sl, 0       # thiet lap bien dem ve 0
loop:   # thuc hien in mot dong cua chu E
        li      $v0, 11
        lb      $a0, 0($s0)
        syscall
        addi    $sl, $sl, 1
        addi    $s0, $s0, 1
        bne     $sl, 16, loop   # in den khi nao du 16 ky tu
        addi    $s0, $s0, -36   # chuyen con tro den chu C
        li      $sl, 0          # thiet lap bien dem ve 0
loop2:  # thuc hien in mot dong cua chu C
        li      $v0, 11
        lb      $a0, 0($s0)
        syscall
        addi    $sl, $sl, 1
        addi    $s0, $s0, 1
        bne     $sl, 20, loop2  # in den khi nao du 20 ky tu
        addi    $s0, $s0, -42   # chuyen con tro den chu D
        li      $sl, 0          # thiet lap bien dem ve 0
loop3:  # thuc hien in mot dong cua chu D
        li      $v0, 11
        lb      $a0, 0($s0)
        syscall
        addi    $sl, $sl, 1
        addi    $s0, $s0, 1
        bne     $sl, 22, loop3  # in den khi nao du 22 ky tu
        addi    $s0, $s0, 79    # chuyen con tro den chu E cua dong tiep theo
        li      $sl, 0          # thiet lap bien dem ve 0
        addi    $s6, $s6, -1    # so dong -1
        li      $v0, 11
        li      $a0, 10
        syscall
        # xuong dong
        bne     $s6, 0, loop
        j       end             # in du 16 dong thi ket thuc
#--------------------bao loi----------------------#
```

- Loop : ví dụ ở đây con trỏ bắt đầu từ ô đầu tiên của vùng nhớ cũng tức là ký tự đầu tiên của chữ D dòng 1, vì vậy ta cộng vào con trỏ là 42 tức là Đầu của chữ E và bắt đầu in hết dòng 1 của chữ E
- Loop2 : vì chữ E có 16 ký tự vì vậy khi in hết ta lại trừ con trỏ đi 36 để nó quay lại chữ C và in hết dòng 1 của chữ C
- Loop3 : vì chữ C có 20 ký tự vì vậy khi in hết ta lại trừ con trỏ đi 42 để nó quay lại chữ D và in hết dòng 1 của chữ D
- Cuối dùng con trỏ cộng 79 để nhảy đến ký tự đầu tiên của chữ E dòng 2, và lặp lại từ loop 1 cho đến khi in hết hình

4. Đổi màu cho chữ

- vì màu chữ được lưu trong các thành ghi t1, t2, t3 nên ta chỉ cần yêu cầu người dùng nhập các số mà họ muốn và lưu số đó vào các thanh ghi tương ứng là được

```
#------------------doi mau chu--------------------#
        li      $v0, 4
        la      $a0, m7
        syscall
        # mau cua D(0-9):
        li      $v0, 5
        syscall
        addi    $t1, $v0, 48
        # luu mau cua D vao t1
        bge     $t1, 58, error2
        ble     $t1, 47, error2
        # kiem tra ky tu khong hop le


        li      $v0, 4
        la      $a0, m8
        syscall
        # mau cua C(0-9):
        li      $v0, 5
        syscall
        addi    $t2, $v0, 48
        # luu mau cua C vao t1
        bge     $t2, 58, error2
        ble     $t2, 47, error2
        # kiem tra ky tu khong hop le


        li      $v0, 4
        la      $a0, m9
        syscall
        # mau cua E(0-9):
        li      $v0, 5
        syscall
        addi    $t3, $v0, 48
        # luu mau cua E vao t1
        bge     $t3, 58, error2
        ble     $t3, 47, error2
        # kiem tra ky tu khong hop le
        j create_image
```

5. exit

- chương trình sẽ nhảy đến exit và kết thúc chương trình

```
exit:
        li      $v0, 10
        syscall
        # thoat chuong trinh
```

B. Nguyễn Đức Đại Dương – Bài 8

I. Phân tích bài toán và cách thực hiện:

1. Phân tích đề bài

- Nhập chuỗi ký tự có độ dài là bội của 8 rồi tiến hành lưu dữ liệu vào 3 disk theo hệ thống RAID5
- Block 4 bytes đầu sẽ lưu vào disk1, block 4 bytes tiếp theo sẽ lưu vào disk 2, dữ liệu lưu vào disk3 là 4 bytes parity được tính từ 2 block đầu tiên theo toán tử XOR. 8 bytes tiếp theo lưu lần lượt vào disk1 và disk 3, disk 2 lại được tính theo toán tử XOR từ 2 block trước. Cuối cùng là 8 bytes tiếp sẽ lưu lần lượt vào disk 2 và disk 3, disk 1 lại được tính theo toán tử XOR từ 2 block trước.
- Cứ lần lượt theo chu kỳ như vậy đến khi kết thúc xâu

2. Xây dựng thuật toán

- Nhập một xâu, kiểm tra độ dài xem hợp lệ chưa
- Thực hiện tách từng bytes đẻ xử lí đứa vào disk
- Xây dựng chương trình con để lấy mã HEX khi thực hiện toán tử XOR
- Khai báo 1 mảng để lưu các phần tử parity của từng disk
- Chia làm 3 part như đã phân tích trên đề bài

II. Mã Nguồn

```
.data
     mes: .asciiz "Nhap xau: "
     disk1: .space 4
     disk2: .space 4
     disk3: .space 4
     array: .space 32
     str: .space 1000
     error: .asciiz "Do dai xau khong hop le!\n"
     comma: .asciiz ","
     enter: .asciiz "\n"
     hex: .byte
'0','1','2','3','4','5','6','7','8','9','a','b','c','d','e','f'
     #hex[0] = 0,..., hex[15] = f
     r1: .asciiz "     Disk 1                Disk 2
Disk 3\n"
     r2: .asciiz
"----------------       ----------------       ----------------\n
"
     r3: .asciiz "|      "
     r4: .asciiz "     |       "
     r5: .asciiz "[[ "
     r6: .asciiz "]]       "
     try: .asciiz "Ban muon tiep tuc?"
```

```
.text
      la $s1, disk1 #luu dia chi disk1
      la $s2, disk2
      la $s3, disk3
      la $a2, array    #luu dia chi mang parity
input:
      li $v0, 4
      la $a0, mes
      syscall          #in ra mes
      li $v0, 8
      la $a0, str      #nhap vao xau
      li $a1, 1000
      syscall
      move $s0, $a0    #luu dia chi xau vao $S0

#Kiem tra do dai xau co chia het cho 8
length:
      addi $t0, $zero, 0    #$t0 = length
      addi $t1, $zero, 0    #$t1 = index

count:
      add $t2, $s0, $t1     #luu dia chi cua str[i] vao t2
      lb $t3, 0($t2)        # t3 = str[i]
      nop
      beq $t3, '\n', test   #gap ki tu xuong dong thi nhay xuong
nhan test

      add $t0, $t0, 1       #neu khong thi tang length len 1
      add $t1, $t1, 1       # tang index len 1
      j count

test:
      li $t5, 8
      move $t4, $t0
      beq $t4, 0, not_avai  #xau rong khong hop le
      div $t4, $t5     #chia length cho 8
      mfhi $t2   #luu phan du vao $t2
      beqz $t2, avai   #du bang 0 thi hop le
not_avai:
      li $v0, 4
      la $a0, error
      syscall
      j input

avai:
      j main
```

```
#---------------Chuong trinh con lay ma hexa (parity)--------
parity:
    li $t9, 1            # khoi tao $t9 = 1

loopParity:
    blt $t9, $0, endParity  # Neu $t9 < 0 thi nhay den endParity
    sll $s7, $t9, 2         # $s7 = $t9 << 2 ($s7 = $t9 * 4)
    srlv $a0, $t8, $s7      # $a0 = $t8 >> $s7 (dich phai $t8
theo $s7 bits)
    andi $a0, $a0, 0x0000000f  # $a0 = $a0 & 0x0000000f ( lay
byte cuoi cua $s0 lam so hexa)
    la $t7, hex             # $t7 = dia chi cua mang hex ( mang
cac chu so hexa)
    add $t7, $t7, $a0       # $t7 = $t7 + $a0 (dia chi cua chu so
hexa can lay)
    lb $a0, 0($t7)          # $a0 = *(char*)($t7 + 0) (lay chu so
hexa tu bo nho va dat vao $a0)
    li $v0, 11              # in ra ky tu
    syscall
    add $t9, $t9, -1        # $t9 = $t9 - 1
    j loopParity            # lap lai loopParity

endParity:
    jr $ra                  # Trá°£ vá»  tá»« hÃ m

#------------Ket thuc chuong trinh con-------------------------

#Chuong trinh chinh
main:
     li $v0, 4
     la $a0, r1
     syscall
     li $v0, 4
     la $a0, r2
     syscall
#Xet 6 lan

#----Lan 1 luu block 4 byte vao Disk1, 4 byte tiep vÃ o Disk2,
parity vao Disk3
#xu li ki tu
part1:
     addi $t1, $zero, 0       # bo dem vong lap cho 4 ki tu dau
     addi $t5, $zero, 0       # bo dem de in ra Disk1 lan 1
     addi $t8, $zero, 0       # bo dem de in ra Disk2 lan 1
     la $s1, disk1            # luu dia chi cua Disk1 vao $s1
     la $s2, disk2            # luu dia chi cua Disk2 vao $s2
     la $a2, array            # luu dia chi cua array vao $a2
```

```
part1_1:
      lb $t2, ($s0)       # luu byte tai dia chi $s0 vÃ o $t2
      addi $t0, $t0, -1   # giam $t0 Ä'i 1
      sb $t2, ($s1)       # luu gia tri cua $t2 vao dia chi $s1
(Disk1)
part1_2:
      addi $s4, $s0, 4    # tang dia chi $s0 len 4 va luu vao $s4
      lb $t3, ($s4)       # luu byte tai dia chi $s4 vÃ o $t3
      addi $t0, $t0, -1   # giam $t0 Ä'i 1
      sb $t3, ($s2)       # luu gia tri cua $t3 vao dia chi $s2
(Disk2)
part1_3:
      xor $a3, $t2, $t3   # OR hai ky tu trong $t2 va $t3 luu vao
$a3
      sb $a3, ($a2)       # Luu gia tri cua $a3 vao dia chi $a2
(array)
      addi $a2, $a2, 4
      addi $t1, $t1, 1
      addi $s0, $s0, 1
      addi $s1, $s1, 1
      addi $s2, $s2, 1
      bgt $t1, 3, clear1
      j part1_1
clear1:
      la $s1, disk1
      la $s2, disk2
#in ki tu tu Disk1
      li $v0, 4           #in dinh dang
      la $a0, r3
      syscall

printPart1Disk1:
      lb $a0, ($s1)
      li $v0, 11
      syscall
      addi $t5, $t5, 1
      addi $s1, $s1, 1
      bgt $t5, 3, print1_1
      j printPart1Disk1

print1_1:
      li $v0, 4           #in dinh dang
      la $a0, r4
      syscall

 #in ki tu tu Disk2
      li $v0, 4
      la $a0, r3
      syscall
```

```
printPart1Disk2:
      lb $a0, ($s2)
      li $v0, 11
      syscall
      addi $t8, $t8, 1
      addi $s2, $s2, 1
      bgt $t8, 3, print1_2
      j printPart1Disk2

print1_2:
      li $v0, 4
      la $a0, r4
      syscall


#in parity tu Disk 3
      li $v0, 4
      la $a0, r5
      syscall
      la $a2, array
      addi $t5, $zero, 0

printPart1Disk3:
      lb $t8, ($a2)       #Tai word tai dia chi $a2 vao $t8
      jal parity          # Goi ham con parity de in ra ma parity
      li $v0, 4
      la $a0, comma
      syscall
      addi $t5, $t5, 1    #in 3 parity co dau phay
      addi $a2, $a2, 4
      bgt $t5, 2, end1
      j printPart1Disk3
end1:                   #in parity cuoi
      lb $t8, ($a2)
      jal parity
      li $v0, 4
      la $a0, r6
      syscall
      li $v0, 4
      la $a0, enter
      syscall
      beq $t0, 0, exit1

#----Lan 2 luu block 4 byte vao Disk1, 4 byte tiep vÃ o Disk3,
parity vao Disk2
#xu li ki tu
part2:
      la $s1, disk1
      la $s3, disk3
      la $a2, array
      addi $s0, $s0, 4
      addi $t1, $zero, 0
```

```
part2_1:
      lb $t2, ($s0)       # luu byte tai dia chi $s0 vÃ o $t2
      addi $t0, $t0, -1   # giam $t0 Ä'i 1
      sb $t2, ($s1)       # luu gia tri cua $t2 vao dia chi $s1
(Disk1)
part2_2:
      addi $s4, $s0, 4    # tang dia chi $s0 len 4 va luu vao $s4
      lb $t3, ($s4)       # luu byte tai dia chi $s4 vÃ o $t3
      addi $t0, $t0, -1   # giam $t0 Ä'i 1
      sb $t3, ($s3)       # luu gia tri cua $t3 vao dia chi $s2
(Disk3)
part2_3:
      xor $a3, $t2, $t3
      sb $a3, ($a2)
      addi $a2, $a2, 4
      addi $t1, $t1, 1
      addi $s0, $s0, 1
      addi $s1, $s1, 1
      addi $s3, $s3, 1
      bgt $t1, 3, clear2
      j part2_1
clear2:
      la $s1, disk1
      la $s3, disk3
      addi $t5, $zero, 0

#in ki tu Disk1
      li $v0, 4
      la $a0, r3
      syscall
printPart2Disk1:
      lb $a0, ($s1)
      li $v0, 11
      syscall
      addi $t5, $t5, 1
      addi $s1, $s1, 1
      bgt $t5, 3, print2_1
      j printPart2Disk1

print2_1:
      li $v0, 4
      la $a0, r4
      syscall


#in parity tu Disk2
      la $a2, array
      addi $t5, $zero, 0
      li $v0, 4
      la $a0, r5
      syscall
```

```
printPart2Disk2:
      lb $t8, ($a2)
      jal parity
      li $v0, 4
      la $a0, comma
      syscall
      addi $t5, $t5, 1
      addi $a2, $a2, 4
      bgt $t5, 2, continue2
      j printPart2Disk2

continue2:
      lb $t8, ($a2)
      jal parity
      li $v0, 4
      la $a0, r6
      syscall
      addi $t8, $zero, 0

#in ki tu tu Disk3
      li $v0, 4
      la $a0, r3
      syscall

printPart2Disk3:
      lb $a0, ($s3)
      li $v0, 11
      syscall
      addi $t8, $t8, 1
      addi $s3, $s3, 1
      bgt $t8, 3, end2
      j printPart2Disk3

 end2:
      li $v0, 4
      la $a0, r4
      syscall
      li $v0, 4
      la $a0, enter
      syscall
      beq $t0, 0, exit1

 #----Lan 3 luu block 4 byte vao Disk2, 4 byte tiep vÃ o Disk3,
 parity vao Disk1
 #xu li ki tu
 part3:
      la $s2, disk2
      la $s3, disk3
      la $a2, array
      addi $s0, $s0, 4
      addi $t1, $zero, 0
```

```
part3_1:
      lb $t2, ($s0)       # luu byte tai dia chi $s0 vÃ o $t2
      addi $t0, $t0, -1   # giam $t0 Ä‘i 1
      sb $t2, ($s2)       # luu gia tri cua $t2 vao dia chi $s1
(Disk2)
part3_2:
      addi $s4, $s0, 4    # tang dia chi $s0 len 4 va luu vao $s4
      lb $t3, ($s4)       # luu byte tai dia chi $s4 vÃ o $t3
      addi $t0, $t0, -1   # giam $t0 Ä‘i 1
      sb $t3, ($s3)       # luu gia tri cua $t3 vao dia chi $s2
(Disk3)
part3_3:
      xor $a3, $t2, $t3
      sb $a3, ($a2)
      addi $a2, $a2, 4
      addi $t1, $t1, 1
      addi $s0, $s0, 1
      addi $s2, $s2, 1
      addi $s3, $s3, 1
      bgt $t1, 3, clear3
      j part3_1
clear3:
      la $s2, disk2
      la $s3, disk3
      addi $t5, $zero, 0
#in parity Disk1
      li $v0, 4
      la $a0, r5
      syscall

printPart3Disk1:
      lb $t8, ($a2)
      jal parity       # In ma parity
      li $v0, 4
      la $a0, comma
      syscall
      addi $t5, $t5, 1
      addi $a2, $a2, 4
      bgt $t5, 2, continue3 # In 3 ma dau tien
      j printPart3Disk1

continue3:
      lb $t8, ($a2)
      jal parity
      li $v0, 4
      la $a0, r6
      syscall
```

```
#in ki tu tu Disk 2
      li $v0, 4
      la $a0, r3
      syscall

      addi $t5, $zero, 0

printPart3Disk2:
      lb $a0, ($s2)
      li $v0, 11
      syscall
      addi $t5, $t5, 1
      addi $s2, $s2, 1
      bgt $t5, 3, print3_1
      j printPart3Disk2

#In ki tu tu Disk3
print3_1:
      li $v0, 4
      la $a0, r4
      syscall
      li $v0, 4
      la $a0, r3
      syscall

      addi $t5, $zero, 0
printPart3Disk3:
      lb $a0, ($s3)
      li $v0, 11
      syscall
      addi $t5, $t5, 1
      addi $s3, $s3, 1
      bgt $t5, 3, end3
      j printPart3Disk3

end3:
      li $v0, 4
      la $a0, r4
      syscall
      li $v0, 4
      la $a0, enter
      syscall
      beq $t0, 0, exit1
#-----Het mot luot-----
#-----Cac luot tiep theo-----
nextPart:
      addi $s0, $s0, 4
      j part1
```

```
exit1:
      li $v0, 4
      la $a0, r2
      syscall
      j ask
#----hoi co tiep tuc nhap xau moi khong-----
ask:
      li $v0, 50
      la $a0, try
      syscall
      beq $a0, 0, clear
      j exit

# dua string ve trang thai ban dau de thuc hien lai qua trinh
clear:     la $s0, str
      li $t2, 0
goAgain:
      sb $t2, ($s0)          # set byte o dia chi str ve 0
      addi $s0, $s0, 1
      bge $s0, $t4, input

      j goAgain

#---------------------------------------
end-----------------------------------#

exit: li $v0, 10
      syscall
```

III. Thực thi:

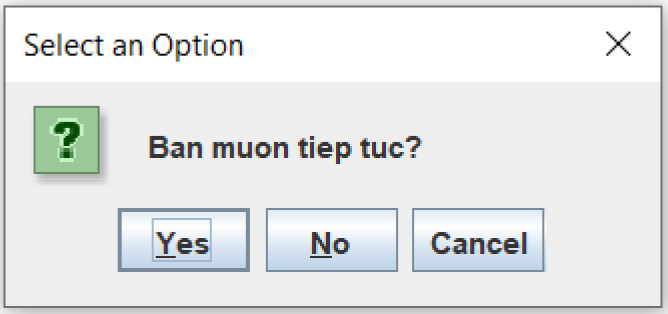
1. Cách chạy chương trình:

- Nhập 1 chuỗi ký tự từ bán phím sao cho độ dài chuỗi ký tự được nhập phải chia hết cho 8 và khác rỗng. Nếu nhập không đúng thì chương trình sẽ báo lỗi và yêu cầu người dùng nhập lại đến khi đúng thì thôi
- Khi nhập đúng, chương trình sẽ chạy và in ra giả lập ổ đĩa RAID 5.
- Khi chương trình chạy và in xong, lựa chọn thực hiện lại chương trình hoặc kết thúc chương trình

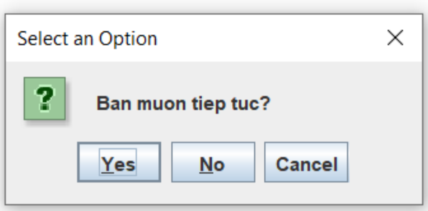
2. Kết quả thực thi:

- **Trường hợp ví dụ đề bài:**

```
Nhap xau: DCE.****ABCD1234HUSTHUST
     Disk 1                 Disk 2                 Disk 3
----------------       ----------------       ----------------
|      DCE.     |      |      ****     |      [[ 6e,69,6f,04]]
|      ABCD     |      [[ 70,70,70,70]]       |      1234     |
[[ 00,00,00,00]]       |      HUST     |      |      HUST     |
----------------       ----------------       ----------------
```




```
Nhap xau: DCE.****ABCD1234HUSTHUSTDCE.****ABCD1234HUSTHUSTDCE.****ABCD1234HUSTHUSTDCE.****ABCD1234HUSTHUST
     Disk 1                 Disk 2                 Disk 3
----------------       ----------------       ----------------
|      DCE.     |      |      ****     |      [[ 6e,69,6f,04]]
|      ABCD     |      [[ 70,70,70,70]]       |      1234     |
[[ 00,00,00,00]]       |      HUST     |      |      HUST     |
|      DCE.     |      |      ****     |      [[ 6e,69,6f,04]]
|      ABCD     |      [[ 70,70,70,70]]       |      1234     |
[[ 00,00,00,00]]       |      HUST     |      |      HUST     |
|      DCE.     |      |      ****     |      [[ 6e,69,6f,04]]
|      ABCD     |      [[ 70,70,70,70]]       |      1234     |
[[ 00,00,00,00]]       |      HUST     |      |      HUST     |
|      DCE.     |      |      ****     |      [[ 6e,69,6f,04]]
|      ABCD     |      [[ 70,70,70,70]]       |      1234     |
[[ 00,00,00,00]]       |      HUST     |      |      HUST     |
----------------       ----------------       ----------------
```



- **Trường hợp nhập xâu có độ dài không chia hết cho 8:**

```
Nhap xau: thuchanhkientrucmaytinh
Do dai xau khong hop le!
Nhap xau: abc
Do dai xau khong hop le!
Nhap xau: Duongne1234567
Do dai xau khong hop le!
Nhap xau:
```

- **Trường hợp nhập xâu rỗng:**

```
Nhap xau:
Do dai xau khong hop le!
Nhap xau:
Do dai xau khong hop le!
Nhap xau:
```

- **Trường hợp nhập xâu ngẫu nhiên có độ dài chia hết cho 8**

```
Nhap xau: daiduong
     Disk 1                 Disk 2                 Disk 3
----------------       ----------------       ----------------
|      daid      |       |      uong      |       [[ 11,0e,07,03]]
----------------       ----------------       ----------------
Nhap xau: DaiDuong_danghocthuchanhkientrucmaytinh!
     Disk 1                 Disk 2                 Disk 3
----------------       ----------------       ----------------
|      DaiD      |       |      uong      |       [[ 31,0e,07,23]]
|      _dan      |       [[ 38,0c,0e,0d]]       |      ghoc      |
[[ 00,00,00,00]]       |      thuc      |       |      hanh      |
|      kien      |       |      truc      |       [[ 1f,1b,10,0d]]
|      mayt      |       [[ 04,0f,11,55]]       |      inh!      |
----------------       ----------------       ----------------
```

-