TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

KIẾN TRÚC MÁY TÍNH

****

MID-TERM MINI PROJECT

GIÁO VIÊN HƯỚNG DẪN : ThS. LÊ BÁ VUI

SINH VIÊN THỰC HIỆN :

TRẦN MINH QUANG 20176858

PHẠM MINH HIẾU 20176758

Mục Lục

Nội dung **Trang**

Bài 3

Sinh viên thực hiện: Phạm Minh Hiếu

Đề bài: Create a program to convert from number to text, in English or Vietnamese (choice 1 of 2).The number in range from 0 to 999 999 999

For example:

Input: 1432

Output: one thousand four hundred and thirty two

Thuật toán:

1/ Nhập dữ liệu đầu vào và kiểm tra dữ liệu

2/ Xử lý:

B1: Tách lớp

+ Tách lớp triệu:

Chia số đầu vào cho 1 000 000. Thương nhận được là lớp triệu. Dư chứa lớp nghìn và lớp đơn vị.

+Tách lớp nghìn và đơn vị

Chia số đầu vào cho 1 000. Thương nhận được là lớp nghìn. Dư chứa lớp đơn vị.

B2: Kiểm tra chữ số có nghĩa

Nếu biến đánh dấu tìm được chữ số có nghĩa lưu ở $t3 =0 thì kiểm tra chữ số có nghĩa

Chia lớp đầu vào cho 100. Nếu thương thu được bằng 0 => in từ hàng chục

Chia lớp đầu vào cho 10. Nếu thương thu được bằng 0 => chỉ in hàng đơn vị.

B3: In từng lớp một

Tách hàng: Chia giá trị của lớp cho 100 thương => hàng trăm,

Lấy dư thu được chia cho 10, thương => hàng chục, dư => hàng đơn vị

In từng chữ số trong lớp => In tên lớp

Khi chữ số đầu tiên được in ra màn hình $t3 =1.

Ý nghĩa của các thanh ghi

$s0: Lưu số n nhập vào, lưu dư trong quá trình tách lớp

$s2: Lưu thương trong quá trình tách lớp

$t6: Giá trị của chữ số sẽ được in

$t0: Đánh dấu lớp đang được đọc

$t3: Đánh dấu việc gặp chữ số có nghĩa

$t1: Lưu kết quả so sánh

$t7: Lưu case chữ số tương ứng

Ý nghĩa của các hàm:

Doc_lop_trieu: Đọc lớp triệu

Doc_lop_Nghin: Đọc lớp nghìn

Doc_lop_Donvi: Đọc lớp đơn vị

In_trieu: In chữ Triệu

In_nghin: In chữ Nghìn

check: Kiểm tra chữ số có nghĩa

In_tram: In chữ số hàng trăm

In_chuc: In chữ số hàng chục

In_donvi: In chữ số hàng đơn vị

in_linh: In chữ linh

in_muoi: In chữ mười

Back_doc_lop: In ten lớp vừa đọc hoặc thoát chương trình

case_0 -> case_9: chuyển số thành chữ

Mã nguồn:

```
.data
# Noi dung input
Message: .asciiz "Nhap so nguyen tu 0 den 999 999 999:"
# Doc so
Message1: .asciiz "mot "
Message2: .asciiz "hai "
Message3 : .asciiz "ba "
Message4: .asciiz "bon "
Message5: .asciiz "nam "
Message6: .asciiz "sau "
Message7: .asciiz "bay "
Message8: .asciiz "tam "
Message9: .asciiz "chin "
Message0: .asciiz "khong "
Message_linh: .asciiz "linh "
Message_muoi: .asciiz "muoi "
```

```
Mess_ch: .asciiz "muoi "
Mess_tr: .asciiz "tram "
Mess_trieu: .asciiz "trieu "
Mess_nghin: .asciiz "nghin "

Mess_er1: "Du lieu nhap vao khong the doc duoc. Moi ban nhap so nguyen tu 0 den 999 999 999"
Mess_er2: "Ban chua nhap du lieu.\nHay nhap mot so nguyen"
Mess_er3: "Ngoai pham vi tinh toan.\nHay nhap mot so nguyen lon hon 0"
Mess_er4: "Ngoai pham vi tinh toan.\nHay nhap mot so nguyen nho hon 999 999 999"

.text
#############################################################################
# Y nghia cac thanh ghi

# $s0: Luu so n nhap vao, luu du trong qua trinh tach lop
# $s2: Luu thuong trong trong qua trinh tach lop
# $t6: Giá tri cua chu so se duoc in
# $t0: danh dau lop dang duoc doc
# $t3: danh dau viec gap chu so co nghia
# $t1: luu ket qua so sanh
# $t7: Luu case chu so tuong ung

# THUAT TOAN
# Tach lop trieu
#       Chia so dau vao cho 1 000 000. th??ng => lop Trieu, du => Lop nghin va lop don vi
#       Kiem tra chu so co nghia
#       In lop Trieu
# Tach lop Nghin
#       Chia du thu duoc cho 1 000. thuong => lop nghin, du =>  lop don vi
#       Kiem tra chu so co nghia
```

```
#       In lop Nghin
# Kiem tra chu so co nghia
#       Neu bien danh dau tim duoc chu so co nghia luu o $t3 =0 thì kiem tra chu so co nghia
#       Chia lop dau vao cho 100. Neu thuong thu duoc bang 0 => in tu hàng chuc
#       Chia lop dau vào cho 10. Neu thuong thu duoc bang 0 => chi in hàng don vi.
# In lop don vi
# Thuat toan in:
#       Tach hang: Chia gia tr? cua lop cho 100 thuong => hang tram,
#               roi lay du thu duoc chia cho 10, thuong => hang chuc, du => hang don vi
#       In tung hang, moi lan in 1 chu so thì goi case tuong ung
#
#################################################################################
main:
# Input so n luu vao thanh ghi $s0
        li $v0, 51
        la $a0, Message
        syscall
#Check loi
        beq $a1, -1, er1      # Neu nhap sai kieu du lieu, in thong bao và quay lai man hinh nhap so
        beq $a1, -2, done     # An cancel thi thoat chuong trinh
        beq $a1, -3, er2      # Neu khong nhap input, in thong bao và quay lai man hinh nhap so



        bltz $a0, er3         #Neu so nhap vao <0, in thong bao và quay lai man hinh nhap so


        li $t1,999999999
        slt $t0,$t1,$a0
        bne $t0,$zero,er4     #Neu so nhap vao >999 999 999, in thong bao và quay lai man hinh nhap so
```

```
        add $s0, $0, $a0     # n=$s0
        beq $s0, $0, khong# Neu so nhap vao bang 0 thi in ra man hinh
        j Doc_lop_trieu
done:
        li $v0, 10
      syscall
#####################################################################
#Truong hop dau vao bang 0 truc tiep in ra man hinh
khong:
        li $v0, 4
        la $a0, Message0
        syscall
#####################################################################
# thuat toan tach lop
# Chia n cho 1 000 000, Thuong luu vao s2, Du luu vao thanh ghi $s0, $s0 = n mod 1 000
000
# doc ket qua phan thuong (lop trieu)
# Chia n = $s0 cho 1000, Thuong luu vao s2 , Du luu vao thanh ghi $s0, $s0 = n mod 1
000
# doc ket qua phan thuong (lop nghin)
# doc ket qua thanh ghi $s0 (lop don vi)
#####################################################################
Doc_lop_trieu:
        li $t0, 1                    # Danh dau lop trieu dang xu ly
        div $s2, $s0, 1000000              # lay n/1 000 000, thuong luu vao $s2
        beq $s2,$zero, Doc_lop_Nghin  # Neu khong co lop trieu chuyen sang tach lop
nghin
        mfhi $s0                     # du luu o thanh ghi hi gán vào $s0
        j check                            # kiem tra chu so co nghia


In_trieu:
        li $v0, 4                    #in chu trieu
```

```
        la $a0, Mess_trieu
        syscall

Doc_lop_Nghin:
        li $t0, 2                   # Danh dau lop nghin dang xu ly
        div $s2, $s0, 1000          # lay n/1 000, thuong luu vao $s2
        beq $s2,$zero, Doc_lop_Donvi  # Neu khong co lop nghin chuyen sang tach lop don vi
        mfhi $s0                    # du luu o thanh ghi hi gán vào $s0
        j check                     # kiem tra chu so co nghia

In_nghin:
        li $v0, 4                   # in chu nghin
        la $a0, Mess_nghin
        syscall

Doc_lop_Donvi:
        li $t0, 3                   # Danh dau lop don vi dang xu ly
        beq $s0, $0, done
        add $s2, $s0, $0            # Chuyen lop don vi luu o $s0 vào $s2 de xu ly
        j check                     # kiem tra chu so co nghia
################################################
# Ham kiem tra chu so co nghia
check:
        beq $t3, 1, In_tram         # $t3 = 0 => chua tung in, $t3=1 => in binh thuong khong phai kiem tra chu so co nghia
        div $t1, $s2, 10            # lay $s2 chia 10, thuong luu vao $t1
        beq $t1,$zero, In_donvi     # Neu $t1 = 0 lop co 2 chu so 0 o dau => doc hang don vi
        div $t1, $s2, 100           # lay $s2 chia 100, thuong luu vao $t1
        beq $t1,$zero, In_chuc      # Neu $t1 = 0 lop co co hang tram = 0 => doc hang chuc
        j In_tram                   # Bat dau doc lop hien tai
################################################
# In hang tram, hang chuc, hang don vi
```

```
In_tram:
        div $t6, $s2, 100          # lay n/100, thuong luu vao $t6
        mfhi $s2                   # du luu o thanh ghi hi gán vào $s2
        jal case_0                 # in chu so hang tram
        nop
        li $v0, 4                  # in chu tram
        la $a0, Mess_tr
        syscall
        beq $s2, $0, Back_doc_lop      # neu lop chi co chu hang tram thi doc lop tiep
theo
In_chuc:
        div $t6, $s2, 10           # lay n/100, thuong luu vao $t6
        mfhi $s2                   # du luu o thanh ghi hi gán vào $s2
        beq $t6, $0, in_linh       # neu hang chuc bang 0 => in chu "linh"
        li $t1, 1
        beq $t6, $t1 , in_muoi     # neu hang chuc bang 1 => in chu "muoi"
        jal case_2                 # in chu so hang chuc
        nop
        li $v0, 4                  # in chu muoi (o hang chuc)
        la $a0, Mess_ch
        syscall
In_donvi:
        add $t6, $s2, $0           # in hang don vi luu o $s2
        beq $t6,$zero, Back_doc_lop    # neu hang don vi = 0 => in ten lop hoac ket thuc
chuong trinh
        mfhi $s2                   # du luu o thanh ghi hi gán vào $s2

        jal case_0                 # in chu so hang don vi
        nop
Back_doc_lop:
        li $t3,1                   # kiem tra da in chu so nào ra man hinh chua
        beq $t0,1, In_trieu        # Truong hop cac chu so dang doc o lop trieu => in ten
lop
```

```
        beq $t0,2, In_nghin         # Truong hop cac chu so dang doc o lop nghin => in ten
lop
        beq $t0,3, done             # Truong hop cac chu so dang doc o lop don vi => ket
thuc chuong trinh
in_linh:
        li $v0, 4                   # in chu linh
        la $a0, Message_linh
        syscall
        j In_donvi


in_muoi:
        li $v0, 4                   # in chu muoi
        la $a0, Message_muoi
        syscall
        j In_donvi
#####################################################################
## case doc chu tuong ung voi so
#####################################################################
case_0:
        bne $t6, $0, case_1
        li $v0, 4
        la $a0, Message0
        syscall
        j break_case
case_1:
        addi $t7, $0, 1
        bne $t6, $t7, case_2
        li $v0, 4
        la $a0, Message1
        syscall
        j break_case
case_2:
        addi $t7, $0, 2
```

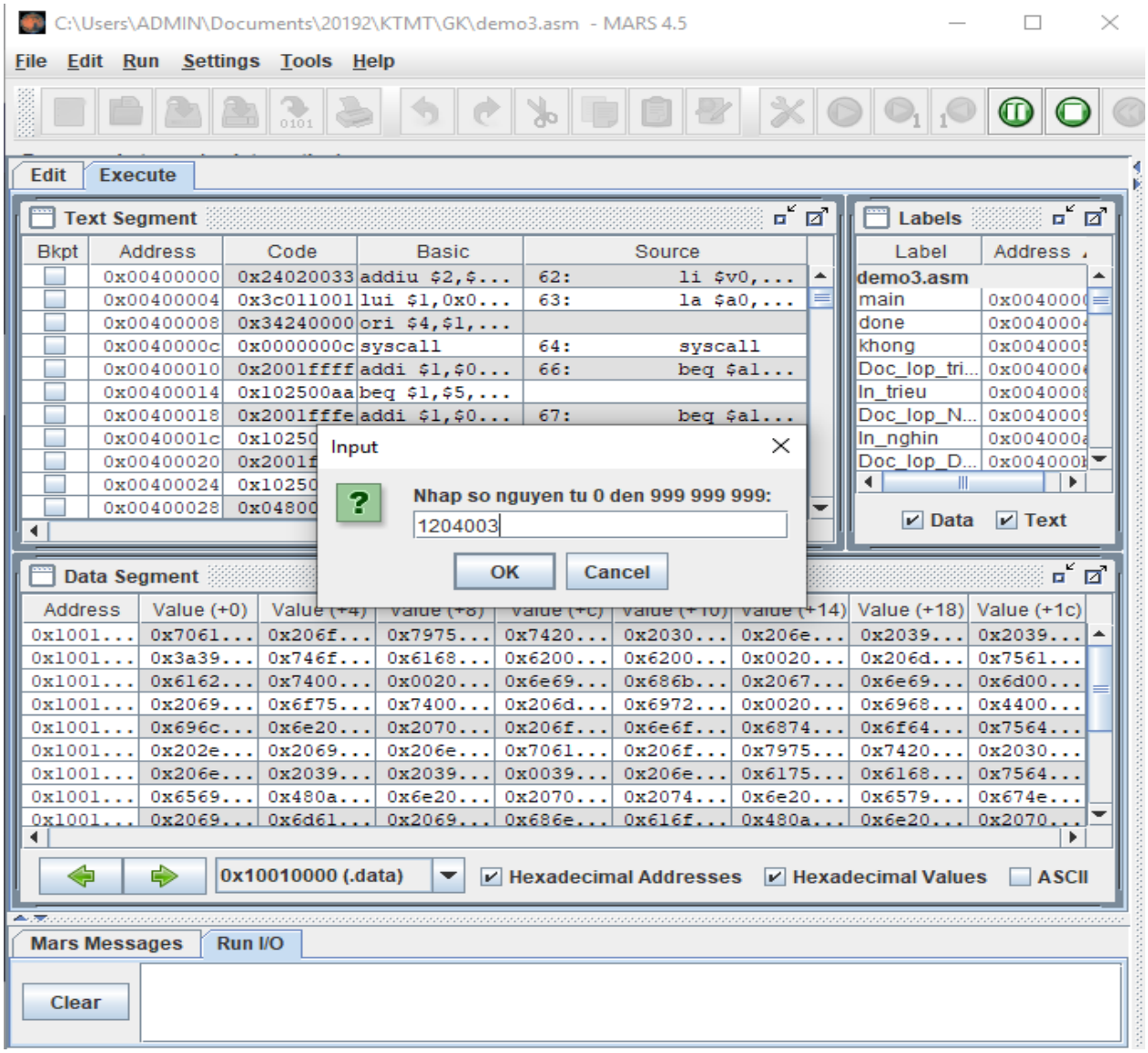
```
        bne $t6, $t7, case_3
        li $v0, 4
        la $a0, Message2
        syscall
        j break_case
case_3:
        addi $t7, $0, 3
        bne $t6, $t7, case_4
        li $v0, 4
        la $a0, Message3
        syscall
        j break_case
case_4:
        addi $t7, $0, 4
        bne $t6, $t7, case_5
        li $v0, 4
        la $a0, Message4
        syscall
        j break_case
case_5:
        addi $t7, $0, 5
        bne $t6, $t7, case_6
        li $v0, 4
        la $a0, Message5
        syscall
        j break_case
case_6:
        addi $t7, $0, 6
        bne $t6, $t7, case_7
        li $v0, 4
        la $a0, Message6
        syscall
```

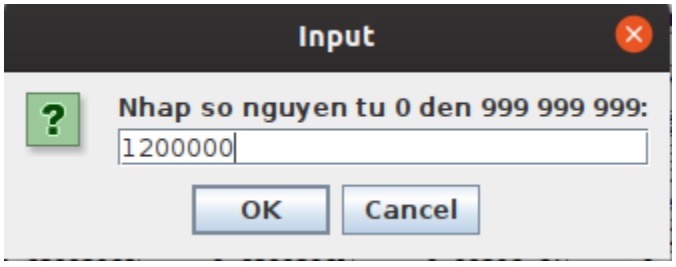
```
        j break_case
case_7:
        addi $t7, $0, 7
        bne $t6, $t7, case_8
        li $v0, 4
        la $a0, Message7
        syscall
        j break_case
case_8:
        addi $t7, $0, 8
        bne $t6, $t7, case_9
        li $v0, 4
        la $a0, Message8
        syscall
        j break_case
case_9:
        li $v0, 4
        la $a0, Message9
        syscall
        j break_case
break_case:
        jr $ra
#####################################################################
# Thong bao loi
er1:
        li $v0, 55
        la $a0, Mess_er1
        syscall
        j main
er2:
        li $v0, 55
        la $a0, Mess_er2
```

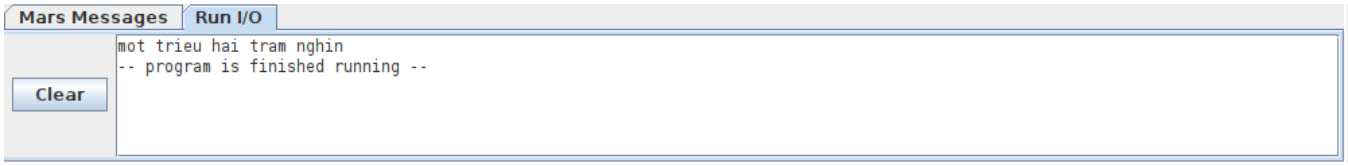
```
        syscall
        j main
er3:
        li $v0, 55
        la $a0, Mess_er3
        syscall
        j main
er4:
        li $v0, 55
        la $a0, Mess_er4
        syscall
j main
```

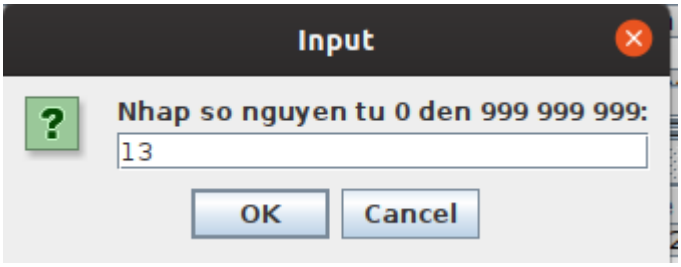
Kết quả:

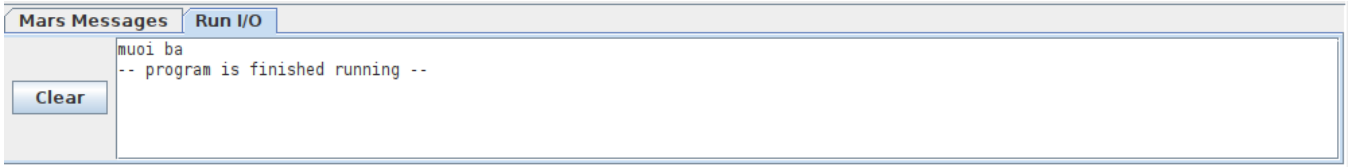
Một số kết quả chạy thử:






Mars Messages
Run I/O
Clear
mot trieu hai tram nghin
-- program is finished running --


Input
Nhap so nguyen tu 0 den 999 999 999:
13
OK
Cancel


Mars Messages
Run I/O
Clear
muoi ba
-- program is finished running --

BÀI 10

SINH VIÊN THỰC HIỆN: TRẦN MINH QUANG

Đề bài: Write a program that gets an integer i from the user and creates the table show below on the screen (example inputs provided). Subroutines are required for power, square, and hexadecimal (in 32-bit arithmetic, attend to overflowed results). Hint: Hexadecimal can be done with shifts and masks because the size is 32 bit.

i	power(2,i)	square(i)	Hexadecimal(i)
10	1024	100	0xA
7	128	49	0x7
16	65536	256	0x10

Thuật Toán:

1. **Nhập và kiểm tra dữ liệu đầu vào.**

   Giới hạn điều kiên từ 1 tới 30, người dùng nhập 0 sẽ thoát chương trình.

2. **Tính lũy thừa cơ số 2 của i (2^i):**

- Ban đầu, khởi tạo thanh ghi ghi kết quả, có giá trị ban đầu là 1
- Vòng lặp for chạy từ j = 0 tới i-1, mỗi lần lặp, nhân thanh ghi ghi kết quả lên 2 và đồng thời giảm j đi một đơn vị.
- Sau khi kết thúc vòng lặp, ta thu được giá trị 2^i

3. **Tính bình phương của i (i^2):**

- Lấy giá trị của i nhân với chính nó, sau đó lưu kết quả vào thanh ghi, ta thu được giá trị i^2

4. **Biểu diễn i về dạng hexa:**

- Sau khi nhận giá trị nhập từ bàn phím. Sẽ sử dụng lệnh sll để chuyển 4 bit sang trái.
- Sau đó sẽ mặt nạ 4 bit này bằng 0xf (0x1111) để lấy 4 bit này.
- Nếu 4 bit này nhỏ hơn hoặc bằng 9 thì sẽ cộng với 48 để lấy mã ASCII từ 0 tới 9
- Nếu 4 bít này lớn hơn hoặc bằng 10 thì sẽ cộng với 55 để lấy mã ASCII từ A tới F

5. **Ý nghĩa các thanh ghi:**

- $s0 chứa giá trị của i
- $s1 chứa giá trị của i^2
- $s2 chứa giá trị của 2^i
- $s3 chứa giá trị của hexa i

Mã nguồn:

#--

```
# $s0 chứa giá trị của i
# $s1 chứa giá trị của i^2
# $s2 chứa giá trị của 2^i
# $s3 chứa giá trị của hexa i
#---------------------------------------------------------------


#---------------------------------------------------------------
# @brief	Nhap tu ban phim mot so nguyen va tra ve cac gia tri binh phuong, hexa, luy thua cua 2
# @param[in]	User input integer i
# @param[out]	power(2,i), square(i), hexadecimal(i)
#---------------------------------------------------------------
.data
str1: .asciiz "Input an interger: "
str2: .asciiz "\nGia Tri i: "
str3: .asciiz "\ni\tpower(2,i)\tsquare(i)\tHexadecimal(i)\n"
str4: .asciiz "\t"
str5: .asciiz "\n"
hex:  .space 10
.text
main:
#---------------------------------------------------------------
	#print: i power(2,i) square(i) Hexadecimal(i)
	li	$v0, 4
	la	$a0, str3
	syscall
Init:
	li	$v0, 51
	la	$a0, str1
	syscall
```

```
        beq     $a1, -1, Init           # if $a1 == -1, nhap lai (kieu nhap vao khong dung)
        beq     $a1, -3, Init           # if $a1 == -3, nhap lai (khong co gia tri nao nhap vao)
        beq     $a1, -2, EXIT           # if $a1 == -2, exit       (nguoi dung cancel)

        move $s0, $a0                                   # i luu vao $s0

        blt     $s0, 0, Init                            # check if i < 0
        bgt     $s0, 30, Init                           # check if i > 30
        beq     $s0, 0, EXIT                            # if i == 0, exit

        j       Power
        nop
H2:
        j       Square
        nop
H3:
        j       Hexadecimal
        nop
H1:
        j       INKETQUA

#-----------------------------------------------------------------
# @label INKETQUA: in bang ket qua duoi man hinh (Run I/O)
#-----------------------------------------------------------------
INKETQUA:

#-----------------------------------------------------------------
        #print i
        li      $v0, 1
```

```
        la      $a0, ($s0)
        syscall
#---------------------------------------------------------------
        # print \t
        la      $a0, str4
        li      $v0, 4
        syscall
#---------------------------------------------------------------
        #print power
        li      $v0, 1
        la      $a0, ($s2)
        syscall
#---------------------------------------------------------------
        # print \t\t
        la      $a0, str4
        li      $v0, 4
        syscall
        syscall
#---------------------------------------------------------------
        #print square
        li      $v0, 1
        la      $a0, ($s1)
        syscall
#---------------------------------------------------------------
        # print \t\t
        la      $a0, str4
        li      $v0, 4
        syscall
        syscall
#---------------------------------------------------------------
```

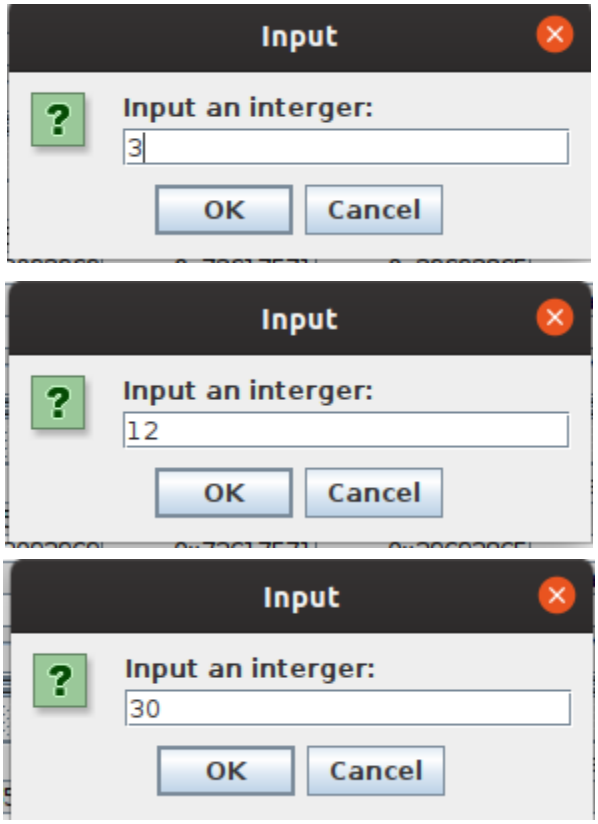
```
        #print Hexadecimal
        li      $v0, 4
        la      $a0, hex
        syscall
#-----------------------------------------------------------------
        #print \n
        la $a0, str5
        li $v0, 4
        syscall
        j       Init
#-----------------------------------------------------------------
# @label Power: tinh gia tri 2^i
# @param[in];       $s0 - gia tri i nhap vao
# @param[out]:  $s2 - gia tri 2^i
#-----------------------------------------------------------------
Power:
        addi    $t1, $zero, 1                   # j = 1
        addi    $s2, $zero, 2                   # power = 1
        addi    $t3, $zero, 2                   # $t3 = 2
for:    slt     $t4, $t1, $s0                   # if i == j
        beq     $t4, $zero, H2                  # thoat khoi vong lap
        mul     $s2, $s2, $t3                   # power *= 2
        mfhi    $s2
        mflo    $s2
        addi    $t1, $t1, 1                     # j += 1
        j       for
#-----------------------------------------------------------------
# @label Square: tinh gia tri i^2
# @param[in]:       $s0 - gia tri i nhap vao
# @param[out]:   $s1 - gia tri i^2
```

```
#---------------------------------------------------------------

Square:
        mult   $s0, $s0
        mfhi   $s1
        mflo   $s1
        j      H3

#---------------------------------------------------------------
# @label Hexadecimal: tra ve gia tri hexa cua so nhap vao
# @param[in]:       $s0 - gia tri i nhap vao
# @param[out]:      $s3 - gia tri hexa cua i
#---------------------------------------------------------------
Hexadecimal:
        la          $a0, hex                    # nap dia chi cua hex vao $a0
        add         $a1, $zero, $s0             # $a1 = i

        li          $t1, 48                     # them 0x vao string hex
        sb          $t1, 0($a0)
        addi        $a0, $a0, 1
        li          $t1, 120
        sb          $t1, 0($a0)
        addi        $a0, $a0, 1

        li          $t0, 8                      # counter = 0
        li          $t2, 0                      # flag $t2=0

hexLoop:
        beqz        $t0, hexDone                # counter == 0 => hexDone
        andi        $t1, $a1, 0xf0000000        # lay 4 bits ngoai cung ben trai
```

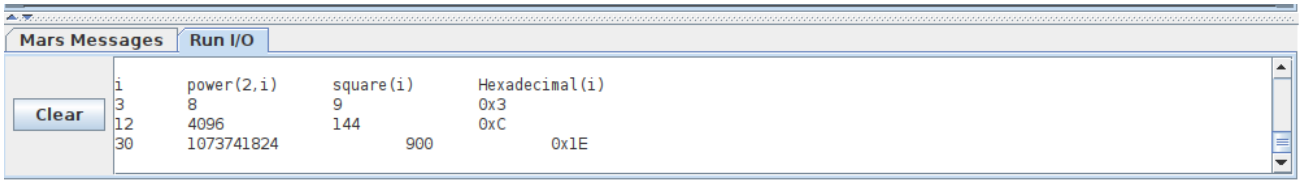
```
        srl     $t1, $t1, 28            # di chuyen 4 bit nay ve ngoai cung
ben phai
        beq     $t1, $t2, continue      # $t1 == $t2 (= 0) => ignore 0
        ble     $t1, 9, less            # $t1 <= 9 => ASCII Code
        addi    $t1, $t1, 55            # [A-F]
        j       writeHex

less:
        addi    $t1, $t1, 48            # [1-9]

writeHex:
        addi    $t2, $t2, -1            # remove flag
        sb      $t1, 0($a0)             # viet ma ASCII vao hex string
        addi    $a0, $a0, 1

continue:
        sll     $a1, $a1, 4             # dich trai 4 bit tiep theo
        addi    $t0, $t0, -1            # counter -= 1
        j       hexLoop

hexDone:
        j       H1
EXIT:
        li      $v0, 10
        syscall
```

Kết quả hiển thị:

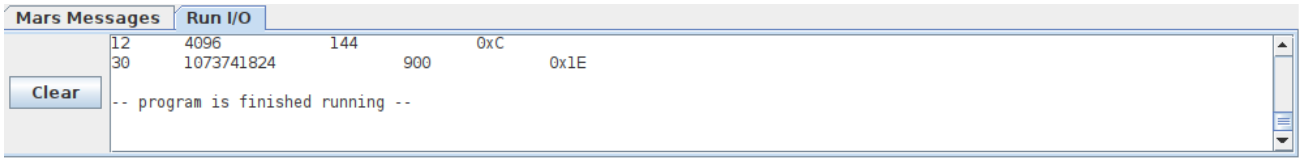
- **Màn hình nhập dữ liệu:**



- **Màn hình kết quả, sau khi chạy các ví dụ trên:**



- **Sau khi nhập vào giá trị 0 hoặc nhấn Cancel thì chương trình kết thúc:**



-Hết-