ÔN TẬP KIẾN TRÚC MÁY TÍNH

Tổng hợp thủ tục và cách dùng.

Cấu trúc chung của chương trình:

.MODEL SMALL

.STACK 100h

.DATA

; Khai báo biến

.CODE

MOV AX, @DATA

MOV DS, AX

; ===========================

; PHẦN THÂN CHƯƠNG TRÌNH

; ===========================

MOV AH, 4Ch

INT 21h

; ===========================

; PHẦN KHAI BÁO THỦ TỤC

END

Code tham khảo: <https://github.com/DalatCoder/KTMT/tree/master/Labs/Lab05>

Tổng hợp các thủ tục và cách dùng

1. Nhập 1 dãy nhị phân

* Input: Không
* Output:

NHAP\_NHI\_PHAN PROC

PUSH AX

PUSH CX

MOV DX, 1

XOR BX, BX

XOR CX, CX

INPUT:

MOV AH, 1

INT 21h

CMP AL, 0Dh

JE BREAK

CMP AL, '0'

JE XULY

CMP AL, '1'

JE XULY

MOV DX, 0

XOR BX, BX

JMP BREAK

XULY:

AND AL, 0Fh

SHL BX, 1

OR BL, AL

INC CX

CMP CX, 16

JB INPUT

BREAK:

POP CX

POP AX

RET

NHAP\_NHI\_PHAN ENDP

* + DX = 0: Xảy ra lỗi trong quá trình nhập
  + DX = 1: Nhập thành công
  + BX = dãy nhị phân vừa nhập
* Cách dùng:
  + Sau khi gọi thủ tục, ta sẽ thu được các giá trị từ thanh ghi BX và DX.
  + Kiểm tra giá trị thanh ghi DX, nếu giá trị là 0 tức là đã xảy ra lỗi trong quá trình nhập, người dùng nhập kí tự khác 0 hoặc 1. Ngược lại, DX mang giá trị 1 tức là việc nhập chuỗi nhị phân thành công
  + Nếu việc nhập thành công, chuỗi nhị phân vừa nhập sẽ được lưu trong thanh ghi BX.
* Đoạn code gọi hàm: trong đoạn code phía dưới, ta tiến hành xuất ***THONGBAO*** nhập 1 dãy nhị phân lên màn hình, gọi hàm ***NHAP\_NHI\_PHAN***, kiểm tra giá trị **DX**, nếu **DX = 0 (JE)** thì ta nhảy về nhãn ***NHAP\_NP*** yêu cầu người dùng nhập lại. Sau khi gọi hàm, chuỗi nhị phân vừa nhập được lưu trong **BX.**

***NHAP\_NP:***

**LEA** DX, ***THONGBAO***

**MOV** AH, 9

**INT** 21h

**CALL** ***NHAP\_NHI\_PHAN***

**CMP** DX, 0

**JE** ***NHAP\_NP***

1. Xuất 1 dãy nhị phân

* Input:
  + BX: Chuỗi nhị phân cần xuất
* Output: Không

XUAT\_NHI\_PHAN PROC

PUSH AX

PUSH CX

PUSH DX

MOV CX, 16

MOV AH, 2

PRINT:

MOV DL, '0'

ROL BX, 1

JNC PRINT\_ZERO:

MOV DL, '1'

PRINT\_ZERO:

INT 21h

LOOP PRINT

POP DX

POP CX

POP AX

RET

XUAT\_NHI\_PHAN ENDP

* Cách dùng:
  + Trước khi gọi thủ tục, ta tiến hành gán chuỗi nhị phân cần xuất vào thanh ghi BX.
  + Gọi thủ tục, chuỗi nhị phân sẽ được xuất ra màn hình
* Đoạn code gọi hàm: trong đoạn code phía dưới, ta tiến hành xuất ***THONGBAO\_XUAT*** để hiển thị dòng thông báo kết quả lên màn hình. Giả sử chuỗi nhị phân cần xuất đang nằm ở biến ***CHUOI\_NP***, ta dùng lệnh **MOV** để gán nó vào **BX,** sau đó gọi hàm ***XUAT\_NHI\_PHAN*** để in chuỗi nhị phân ra màn hình.

**LEA** DX, ***THONGBAO\_XUAT***

**MOV** AH, 9

**INT** 21h

**MOV** BX, CHUOI\_NP

**CALL** ***XUAT\_NHI\_PHAN***

1. Nhập 1 dãy thập lục phân

* Input: Không
* Output:

NHAP\_HEXA PROC

PUSH AX

PUSH CX

XOR BX, BX

XOR CX, CX

MOV DX, 1

INPUT:

MOV AH, 1

INT 21h

CMP AL, 0Dh

JE BREAK

CMP AL, '0'

JB XULY\_LOI

CMP AL, '9'

JBE XULY\_SO

CMP AL, 'A'

JB XULY\_LOI

CMP AL, 'F'

JBE XULY\_CHU

XULY\_LOI:

MOV DX, 0

XOR BX, BX

JMP BREAK

XULY\_SO:

AND AL, 0Fh

JMP XULY

XULY\_CHU:

SUB AL, 37h

XULY:

SHL BX, 4

OR BL, AL

INC CX

CMP CX, 4

JB INPUT

BREAK:

POP CX

POP AX

RET

NHAP\_HEXA ENDP

* + DX = 0: Xảy ra lỗi trong quá trình nhập
  + DX = 1: Nhập thành công
  + BX = dãy thập lục phân vừa nhập
* Cách dùng:
  + Sau khi gọi thủ tục, ta sẽ thu được các giá trị từ thanh ghi BX và DX.
  + Kiểm tra giá trị thanh ghi DX, nếu giá trị là 0 tức là đã xảy ra lỗi trong quá trình nhập, người dùng nhập kí tự khác   
    0-9 và A-F. Ngược lại, DX mang giá trị 1 tức là việc nhập chuỗi thập lục phân thành công
  + Nếu việc nhập thành công, chuỗi thập lục phân vừa nhập sẽ được lưu trong thanh ghi BX.
* Đoạn code gọi hàm: trong đoạn code phía dưới, ta tiến hành xuất ***THONGBAO*** nhập 1 dãy thập lục phân lên màn hình, gọi hàm ***NHAP\_HEXA***, kiểm tra giá trị **DX**, nếu **DX = 0 (JE)** thì ta nhảy về nhãn ***NHAP\_16*** yêu cầu người dùng nhập lại. Sau khi gọi hàm, chuỗi thập lục phân vừa nhập được lưu trong **BX.**

***NHAP\_16:***

**LEA** DX, ***THONGBAO***

**MOV** AH, 9

**INT** 21h

**CALL** ***NHAP\_HEXA***

**CMP** DX, 0

**JE** ***NHAP\_16***

1. Xuất 1 dãy thập lục phân

* Input:
  + BX: Chuỗi thập lục phân cần xuất
* Output: Không

XUAT\_HEXA PROC

PUSH AX

PUSH CX

PUSH DX

MOV CX, 4

PRINT:

MOV DL, BH

SHR DL, 4

CMP DL, 9

JBE XUAT\_SO

JA XUAT\_CHU

XUAT\_SO:

MOV AH, 2

OR DL, 30h

INT 21h

JMP CONTINUE

XUAT\_CHU:

MOV AH, 2

ADD DL, 37h

INT 21h

CONTINUE:

ROL BX, 4

LOOP PRINT

POP DX

POP CX

POP AX

RET

XUAT\_HEXA ENDP

* Cách dùng:
  + Trước khi gọi thủ tục, ta tiến hành gán chuỗi thập lục phân cần xuất vào thanh ghi BX.
  + Gọi thủ tục, chuỗi thập lục phân sẽ được xuất ra màn hình
* Đoạn code gọi hàm: trong đoạn code phía dưới, ta tiến hành xuất ***THONGBAO\_XUAT*** để hiển thị dòng thông báo kết quả lên màn hình. Giả sử chuỗi thập lục phân cần xuất đang nằm ở biến ***CHUOI\_HEX***, ta dùng lệnh **MOV** để gán nó vào **BX,** sau đó gọi hàm ***XUAT\_HEXA*** để in chuỗi thập lục phân ra màn hình.

**LEA** DX, ***THONGBAO\_XUAT***

**MOV** AH, 9

**INT** 21h

**MOV** BX, CHUOI\_HEX

**CALL** ***XUAT\_HEXA***

1. Nhập 1 số thập phân

* Input: Không
* Output:

NHAP\_THAP\_PHAN PROC

PUSH AX

PUSH CX

PUSH SI

XOR BX, BX

INPUT:

MOV AH, 1

INT 21h

CMP AL, 0Dh

JE BREAK

CMP AL, '0'

JB XULY\_LOI

CMP AL, '9'

JA XULY\_LOI

AND AL, 0Fh

MOV CL, AL

XOR CH, CH

MOV AX, BX

MOV SI, 10

MUL SI

MOV BX, AX

ADD BX, CX

JMP INPUT

XULY\_LOI:

XOR BX, BX

MOV DX, 0

JMP EXIT

BREAK:

MOV DX, 1

EXIT:

POP SI

POP CX

POP AX

RET

NHAP\_THAP\_PHAN ENDP

* + DX = 0: Xảy ra lỗi trong quá trình nhập
  + DX = 1: Nhập thành công
  + BX = số thập phân vừa nhập
* Cách dùng:
  + Sau khi gọi thủ tục, ta sẽ thu được các giá trị từ thanh ghi BX và DX.
  + Kiểm tra giá trị thanh ghi DX, nếu giá trị là 0 tức là đã xảy ra lỗi trong quá trình nhập, người dùng nhập kí tự khác 0-9. Ngược lại, DX mang giá trị 1 tức là việc nhập số thập phân thành công
  + Nếu việc nhập thành công, số thập phân vừa nhập sẽ được lưu trong thanh ghi BX.
* Đoạn code gọi hàm: trong đoạn code phía dưới, ta tiến hành xuất ***THONGBAO*** nhập 1 số thập phân lên màn hình, gọi hàm ***NHAP\_THAP\_PHAN***, kiểm tra giá trị **DX**, nếu **DX = 0 (JE)** thì ta nhảy về nhãn ***NHAP\_10*** yêu cầu người dùng nhập lại. Sau khi gọi hàm, số thập phân vừa nhập được lưu trong **BX.**

***NHAP\_10:***

**LEA** DX, ***THONGBAO***

**MOV** AH, 9

**INT** 21h

**CALL** ***NHAP\_THAP\_PHAN***

**CMP** DX, 0

**JE** ***NHAP\_10***

1. Xuất 1 số thập phân:

* Input:
  + AX: Số thập phân cần xuất
* Output: Không

XUAT\_THAP\_PHAN PROC

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV BX, 10

XOR CX, CX

XULY\_STACK:

XOR DX, DX

DIV BX

PUSH DX

INC CX

CMP AX, 0

JA XULY\_STACK

PRINT:

POP DX

OR DX, 30h

MOV AH, 2

INT 21h

LOOP PRINT

POP DX

POP CX

POP BX

POP AX

RET

XUAT\_THAP\_PHAN ENDP

* Cách dùng:
  + Trước khi gọi thủ tục, ta tiến hành gán số thập phân cần xuất vào thanh ghi AX.
  + Gọi thủ tục, số thập phân sẽ được xuất ra màn hình

Đoạn code gọi hàm: trong đoạn code phía dưới, ta tiến hành xuất ***THONGBAO\_XUAT*** để hiển thị dòng thông báo kết quả lên màn hình. Giả sử ta gọi hàm NHAP\_THAP\_PHAN và thu được số thập phân ở BX, để xuất số này ra, ta tiến hành gán BX vào AX và gọi hàm XUAT\_THAP\_PHAN để in số thập phân lên màn hình.

**LEA** DX, ***THONGBAO\_XUAT***

**MOV** AH, 9

**INT** 21h

**MOV** AX, BX

**CALL** ***XUAT\_THAP\_PHAN***

1. Lấy thông tin ngày giờ của hệ thống

* Việc lấy thông tin ngày giờ của hệ thống rất dễ dàng vì đã có hàm ngắt INT 21h, AH = 2Ah. Phần khó nhất là làm sao để xuất được thông tin này ra màn hình ở dạng thập phân, ta đã có thủ tục để xuất 1 số thập phân, lúc này ta chỉ cần gán giá trị thập phân cần xuất vào thanh ghi AX và gọi hàm XUAT\_THAP\_PHAN thì mọi chuyện xem như đã xong.
* Đoạn code lấy thông tin ngày giờ hệ thống:

**MOV** AH, 2Ah

**INT** 21h

Sau khi thực hiện đoạn code trên, ta thu được các giá trị:

* + **CX:** chứa giá trị năm hiện tại
  + **DH:** chứa giá trị tháng hiện tại
  + **DL:** chứa giá trị ngày hiện tại
* Đoạn code xuất thông tin ngày ra màn hình:

**LEA** DX, ***MSG\_NGAY***

**MOV** AH, 9

**INT** 21h

**MOV** AL, DL

**XOR** AH, AH

**CALL** ***XUAT\_THAP\_PHAN***

* Đoạn code xuất thông tin tháng ra màn hình:

**LEA** DX, ***MSG\_THANG***

**MOV** AH, 9

**INT** 21h

**MOV** AL, DH

**XOR** AH, AH

**CALL** ***XUAT\_THAP\_PHAN***

* Đoạn code xuất thông tin năm ra màn hình:

**LEA** DX, ***MSG\_NAM***

**MOV** AH, 9

**INT** 21h

**MOV** AX, CX

**CALL** ***XUAT\_THAP\_PHAN***

* Giải thích code: Trong 3 đoạn code trên, phần đầu tiên là xuất thông báo hiển thị kết quả ra màn hình. Sau đó chúng ta sẽ lần lượt xuất từng giá trị ra màn hình ở dạng thập phân. Với trường hợp xuất năm, ta thấy rất rõ ràng, chỉ cần gán giá trị năm được lưu trong **CX** vào **AX**, sau đó gọi hàm **XUAT\_THAP\_PHAN** là xong. Tuy nhiên với việc xuất ngày và tháng lại hơi rườm rà 1 tý. Với trường hợp tháng, giá trị được lưu trong thanh ghi **DH**, ta không thể thực hiện lệnh gán trực tiếp được bởi vì **DH** là thanh ghi 8bit và **AX** là thanh ghi 16bit. Vì vậy ta cần phải tìm 1 cách nào đó để gán giá trị từ **DH** vào **AX**. Ta có thể thực hiện việc này thông qua lệnh gán giá trị **DH** vào thanh ghi **AL** (8bit). Giả sử hình minh họa dưới đây, **AX** gồm 2 thanh ghi nhỏ hơn là **AH** và **AL**, trước đó chúng ta gọi lệnh lấy thông tin ngày hệ thống nên lúc này giả sử **AH** đang có giá trị là **2A**h, **AL** đang mang giá trị bất kì.

|  |  |
| --- | --- |
| **AX** | |
| **AH** | **AL** |
| 2Ah | 11h |

Giả sử ngày hiện tại là 10 (tức **A**h). Ta thực hiện lệnh gán **DL** vào **AL**, lúc này thanh ghi **AX** sẽ như sau:

|  |  |
| --- | --- |
| **AX** | |
| **AH** | **AL** |
| 2Ah | 0Ah |

Giá trị thanh ghi **AX** lúc này là **2A0A**h, nếu ta gọi hàm **XUAT\_THAP\_PHAN** bây giờ thì giá trị xuất ra sẽ không phải là **10** mà là **10762** trong hệ thập phân. Vì vậy, việc chúng ta cần làm đầu tiên là xóa đi phần bit cao **AH**. Lúc này thanh ghi **AX** sẽ như sau:

|  |  |
| --- | --- |
| **AX** | |
| **AH** | **AL** |
| 00h | 0Ah |

Lúc này thanh ghi **AX** sẽ mang giá trị **A**h (tức 10 trong hệ thập phân). Như vậy khi ta gọi hàm **XUAT\_THAP\_PHAN**, chúng ta sẽ thu được giá trị 10 được hiển thị trên màn hình.

Đến lúc này, ta đã thực hiện được lệnh gán giá trị trong thanh ghi **DL** vào **AX**.

Ta làm tương tự với các trường hợp khác, khi cần xuất giá trị thập phân được chứa trong thanh ghi 8bit, ta thực hiện lệnh gán giá trị đó vào vùng thấp **AL**, sau đó thực hiện lệnh xóa vùng cao **AH** và gọi hàm ***XUAT\_THAP\_PHAN***. Ví dụ khi xuất ngày, ta gán **DL** vào **AL**, xóa **AH** và gọi thủ tục để xuất.

1. Xuất thông tin giờ của hệ thống

Tương tự như việc xuất thông tin ngày tháng, ta chỉ cần gọi hàm ngắt và gọi thủ tục XUAT\_THAP\_PHAN để xuất lần lượt từng giá trị ra màn hình.

* Đoạn code lấy thông tin ngày giờ hệ thống:

**MOV** AH, 2Ch

**INT** 21h

Sau khi thực hiện đoạn code trên, ta thu được các giá trị:

* + **CH:** chứa giá trị giờ hiện tại
  + **CL:** chứa giá trị phút hiện tại
  + **DH:** chứa giá trị giây hiện tại
* Sau khi đã có đủ các giá trị, ta thực hiện gán giá trị cần xuất vào thanh ghi **AX** và gọi thủ tục ***XUAT\_THAP\_PHAN***. Phần giải thích tương tự trong mục 7 phía trên.
* Đoạn code xuất thông tin giờ ra màn hình:

**LEA** DX, ***MSG\_GIO***

**MOV** AH, 9

**INT** 21h

**MOV** AL, CH

**XOR** AH, AH

**CALL** ***XUAT\_THAP\_PHAN***

* Đoạn code xuất thông tin phút ra màn hình:

**LEA** DX, ***MSG\_PHUT***

**MOV** AH, 9

**INT** 21h

**MOV** AL, CL

**XOR** AH, AH

**CALL** ***XUAT\_THAP\_PHAN***

* Đoạn code xuất thông tin giây ra màn hình:

**LEA** DX, ***MSG\_NAM***

**MOV** AH, 9

**INT** 21h

**MOV** AX, DH

**XOR** AH, AH

**CALL** ***XUAT\_THAP\_PHAN***

1. Tính chu vi, diện tích hình chữ nhật

* Công thức:
  + ChuVi = (Dài + Rộng) x 2
  + DienTich = Dai x Rong
* Input:
  + **BH**: Chiều dài
  + **BL**: Chiều rộng
* Output:

; THU TUC TINH CHU VI

; Input:

; - BH: Chieu dai hinh CN

; - BL: Chieu rong hinh CN

; Output:

; - AX: CHU VI HINH CHU NHAT

TINH\_CHU\_VI PROC

PUSH BX

ADD BH, BL

MOV AL, 2

MUL BH

POP BX

RET

TINH\_CHU\_VI ENDP

; ================================

; THU TUC TINH DIEN TICH

; Input:

; - BH: Chieu dai hinh CN

; - BL: Chieu rong hinh CN

; Output:

; - AX: DIEN TICH HINH CN

TINH\_DIEN\_TICH PROC

PUSH BX

MOV AL, BH

MUL BL

POP BX

RET

TINH\_DIEN\_TICH ENDP

* + **AX**: Chu vi hình chữ nhật
* Cách dùng:
  + Trước khi gọi thủ tục, ta tiến hành gán giá trị chiều dài vào thanh ghi **BH**, gán giá trị chiều rộng vào thanh ghi **BL**.
  + Gọi thủ tục, sau đó giá trị chu vi sẽ được chứa trong thanh ghi **AX**.
* Đoạn code gọi hàm: trong đoạn code phía dưới, ta tiến hành xuất ***THONGBAO\_CHUVI*** , hiển thị thông báo kết quả chu vi hình chữ nhật lên màn hình. Giả sử ta có biến **Dai** kiểu **DB** lưu giá trị chiều dài, biến **Rong** kiểu **DB** lưu giá trị chiều rộng. Ta tiến hành gán giá trị **Dai** vào **BH** và gán giá trị **Rong** vào **BL**. Tiến hành gọi hàm, sau khi hàm được gọi, giá trị chu vi sẽ được lưu ở thanh ghi **AX**. Với thủ tục ***XUAT\_THAP\_PHAN***, ta cần gán giá trị cần xuất vào thanh ghi **AX**, tuy nhiên trong trường hợp này, giá trị chu vi đã có sẵn trong thanh ghi **AX** rồi nên ta chỉ cần gọi thủ tục là xong.

**LEA** DX, ***THONGBAO\_CHUVI***

**MOV** AH, 9

**INT** 21h

**MOV** BH, Dai

**MOV** DL, Rong

**CALL** ***TINH\_CHU\_VI***

**CALL** ***XUAT\_THAP\_PHAN***

* Ta làm tương tự khi cần tính diện tích.
* Nâng cao: Ta có thể nhập vào giá trị chiều dài hoặc chiều rộng là 1 số. Hoặc dùng thủ tục NHAP\_THAP\_PHAN để nhập vào 1 số thập phân, mở rộng phạm vi của việc tính toán. Hàm NHAP\_THAP\_PHAN sẽ lưu trữ giá trị thập phân vừa nhập trong thanh ghi BX, ta có thể khai báo hai biến Dai kiểu DW (1 word) và Rong kiểu DW để lưu trữ lần lượt chiều dài và chiều rộng. Không thể khai báo biến kiểu DB vì DB 8bit, trong khi đó BX 16bit, ta không thể gán được. Tiếp theo ta điều chỉnh lại hàm, cho hàm nhận vào giá trị BX là chiều dài, CX là chiều rộng. Hàm tính chu vi và diện tích sẽ lưu trữ các giá trị tương ứng trong thanh ghi AX. Việc xuất giá trị này ra ta làm tương tự như phía trên. Tham khảo code tại: (…)

; THU TUC TINH CHU VI

; Input:

; - BX: Chieu dai hinh CN

; - CX: Chieu rong hinh CN

; Output:

; - AX: CHU VI HINH CHU NHAT

TINH\_CHU\_VI PROC

PUSH BX

ADD BX, CX

MOV AX, 2

MUL BX

POP BX

RET

TINH\_CHU\_VI ENDP

; ================================

; THU TUC TINH DIEN TICH

; Input:

; - BX: Chieu dai hinh CN

; - CX: Chieu rong hinh CN

; Output:

; - AX: DIEN TICH HINH CN

TINH\_DIEN\_TICH PROC

PUSH BX

MOV AX, BX

MUL CX

POP BX

RET

TINH\_DIEN\_TICH ENDP

1. Chương trình tính tổng, hiệu, tích, thương

Để tính tổng ta dùng lệnh ADD, hiệu ta dùng SUB, tích ta dùng MUL và thương là DIV. Dữ liệu đầu vào là 2 số, ta có thể nhập từ bàn phím thông qua lệnh ngắt INT 21h, AH=1, hoặc dùng thủ tục NHAP\_THAP\_PHAN. Ta thực hiện tính toán, và để xuất, ta gọi thủ tục XUAT\_THAP\_PHAN

* Lưu ý: Khi trừ 2 số cho nhau, nếu kết quả âm, ta cần kiểm tra để xuất cho đúng

CODE THAM KHẢO:

.MODEL SMALL

.STACK 100h

.DATA

MSG1 DB 10,13, 'Nhap vao 1 so thap phan: $'

MSG2 DB 10,13, 'So thap phan vua nhap la: $'

.CODE

; LAY DU LIEU TU DATA VAO DS

MOV AX, @DATA

MOV DS, AX

; ====================================

NHAP\_10:

LEA DX, MSG1

CALL XUAT\_CHUOI

CALL NHAP\_THAP\_PHAN

CMP DX, 0

JE NHAP\_10

LEA DX, MSG2

CALL XUAT\_CHUOI

MOV AX, BX

CALL XUAT\_THAP\_PHAN

; ====================================

; DUNG MAN HINH, XEM KET QUA

MOV AH, 4Ch

INT 21h

; ====================================

; KHAI BAO THU TUC

; ====================================

XUAT\_CHUOI PROC

PUSH AX

MOV AH, 9

INT 21h

POP AX

RET

XUAT\_CHUOI ENDP

1. Nhập xuất thập phân:

NHAP\_THAP\_PHAN PROC

PUSH AX

PUSH CX

PUSH SI

XOR BX, BX

INPUT:

MOV AH, 1

INT 21h

CMP AL, 0Dh

JE BREAK

CMP AL, '0'

JB XULY\_LOI

CMP AL, '9'

JA XULY\_LOI

AND AL, 0Fh

MOV CL, AL

XOR CH, CH

MOV AX, BX

MOV SI, 10

MUL SI

MOV BX, AX

ADD BX, CX

JMP INPUT

XULY\_LOI:

XOR BX, BX

MOV DX, 0

JMP EXIT

BREAK:

MOV DX, 1

EXIT:

POP SI

POP CX

POP AX

RET

NHAP\_THAP\_PHAN ENDP

XUAT\_THAP\_PHAN PROC

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV BX, 10

XOR CX, CX

XULY\_STACK:

XOR DX, DX

DIV BX

PUSH DX

INC CX

CMP AX, 0

JA XULY\_STACK

PRINT:

POP DX

OR DX, 30h

MOV AH, 2

INT 21h

LOOP PRINT

POP DX

POP CX

POP BX

POP AX

RET

XUAT\_THAP\_PHAN ENDP

END

1. Xuất thông tin ngày tháng

XUAT\_THAP\_PHAN PROC

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV BX, 10

XOR CX, CX

XULY\_STACK:

XOR DX, DX

DIV BX

PUSH DX

INC CX

CMP AX, 0

JA XULY\_STACK

PRINT:

POP DX

OR DX, 30h

MOV AH, 2

INT 21h

LOOP PRINT

POP DX

POP CX

POP BX

POP AX

RET

XUAT\_THAP\_PHAN ENDP

END

.MODEL SMALL

.STACK 100h

.DATA

MSG\_NGAY DB 10,13, '- Ngay: $'

MSG\_THANG DB 10,13, '- Thang: $'

MSG\_NAM DB 10,13, '- Nam: $'

NGAY DB ?

THANG DB ?

NAM DW ?

.CODE

; KHOI TAO CHUONG TRINH

MOV AX, @DATA

MOV DS, AX

; LAY NGAY/ THANG/ NAM CUA HE THONG

MOV AH, 2Ah

INT 21h

MOV NGAY, DL

MOV THANG, DH

MOV NAM, CX

; Xuat thong bao ngay

MOV DX, OFFSET MSG\_NGAY

CALL XUAT\_CHUOI

XOR AH, AH

MOV AL, NGAY

CALL XUAT\_THAP\_PHAN

; Xuat thong bao thang

MOV DX, OFFSET MSG\_THANG

CALL XUAT\_CHUOI

XOR AH, AH

MOV AL, THANG

CALL XUAT\_THAP\_PHAN

; Xuat thong bao nam

MOV DX, OFFSET MSG\_NAM

CALL XUAT\_CHUOI

MOV AX, NAM

CALL XUAT\_THAP\_PHAN

; Thoat chuong trinh

MOV AH, 4Ch

INT 21h

; ====================================

; PHAN KHAI BAO THU TUC

; ====================================

XUAT\_CHUOI PROC

MOV AH, 9

INT 21h

RET

XUAT\_CHUOI ENDP

1. Xuất thông tin thời gian giờ phút giây

XUAT\_THAP\_PHAN PROC

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV BX, 10

XOR CX, CX

XULY\_STACK:

XOR DX, DX

DIV BX

PUSH DX

INC CX

CMP AX, 0

JA XULY\_STACK

PRINT:

POP DX

OR DX, 30h

MOV AH, 2

INT 21h

LOOP PRINT

POP DX

POP CX

POP BX

POP AX

RET

XUAT\_THAP\_PHAN ENDP

END

.MODEL SMALL

.STACK 100h

.DATA

MSG\_GIO DB 10,13, '- Gio: $'

MSG\_PHUT DB 10,13, '- Phut: $'

MSG\_GIAY DB 10,13, '- Giay: $'

GIO DB ?

PHUT DB ?

GIAY DB ?

.CODE

; KHOI TAO CHUONG TRINH

MOV AX, @DATA

MOV DS, AX

; LAY GIO/ PHUT/ GIAY CUA HE THONG

MOV AH, 2Ch

INT 21h

MOV GIO, CH

MOV PHUT, CL

MOV GIAY, DH

; Xuat thong bao gio cua he thong

MOV DX, OFFSET MSG\_GIO

CALL XUAT\_CHUOI

XOR AH, AH

MOV AL, GIO

CALL XUAT\_THAP\_PHAN

; Xuat thong bao phut cua he thong

MOV DX, OFFSET MSG\_PHUT

CALL XUAT\_CHUOI

XOR AH, AH

MOV AL, PHUT

CALL XUAT\_THAP\_PHAN

; Xuat thong bao giay cua he thong

MOV DX, OFFSET MSG\_GIAY

CALL XUAT\_CHUOI

XOR AH, AH

MOV AL, GIAY

CALL XUAT\_THAP\_PHAN

; Thoat chuong trinh

MOV AH, 4Ch

INT 21h

; ====================================

; PHAN KHAI BAO THU TUC

; ====================================

XUAT\_CHUOI PROC

MOV AH, 9

INT 21h

RET

XUAT\_CHUOI ENDP

1. Tính chu vi, diện tích hình chữ nhật

; Khai bao thu tuc

XUAT\_CHUOI PROC

MOV AH, 9

INT 21h

RET

XUAT\_CHUOI ENDP

NHAP\_THAP\_PHAN PROC

PUSH AX

PUSH CX

PUSH SI

XOR BX, BX

INPUT:

MOV AH, 1

INT 21h

CMP AL, 0Dh

JE BREAK

CMP AL, '0'

JB XULY\_LOI

CMP AL, '9'

JA XULY\_LOI

AND AL, 0Fh

MOV CL, AL

XOR CH, CH

MOV AX, BX

MOV SI, 10

MUL SI

MOV BX, AX

ADD BX, CX

JMP INPUT

XULY\_LOI:

XOR BX, BX

MOV DX, 0

JMP EXIT

BREAK:

MOV DX, 1

EXIT:

POP SI

POP CX

POP AX

RET

NHAP\_THAP\_PHAN ENDP

.MODEL SMALL

.STACK 100h

.DATA

MSG1 DB 10,13, 'Nhap chieu dai: $'

MSG2 DB 10,13, 'Nhap chieu rong: $'

MSG3 DB 10,13, 'Dien tich: $'

MSG4 DB 10,13, 'Chu vi: $'

DAI DW ?

RONG DW ?

.CODE

MOV AX, @DATA

MOV DS, AX

; Nhap chieu dai

NHAP\_DAI:

LEA DX, MSG1

CALL XUAT\_CHUOI

CALL NHAP\_THAP\_PHAN

CMP DX, 0

JE NHAP\_DAI

MOV DAI, BX

; Nhap chieu rong

NHAP\_RONG:

LEA DX, MSG2

CALL XUAT\_CHUOI

CALL NHAP\_THAP\_PHAN

CMP DX, 0

JE NHAP\_RONG

MOV RONG, BX

; Gan gia tri vao BX, CX

MOV BX, DAI

MOV CX, RONG

; Xuat dien tich

LEA DX, MSG3

CALL XUAT\_CHUOI

CALL TINH\_DIEN\_TICH

CALL XUAT\_THAP\_PHAN

; Xuat chu vi

LEA DX, MSG4

CALL XUAT\_CHUOI

CALL TINH\_CHU\_VI

CALL XUAT\_THAP\_PHAN

MOV AH, 4Ch

INT 21h

; Input:

; - BX: Chieu dai

; - CX: Chieu rong

; Output:

; - AX: Dien tich

TINH\_DIEN\_TICH PROC

MOV AX, BX

MUL CX

RET

TINH\_DIEN\_TICH ENDP

; Input:

; - BX: Chieu dai

; - CX: Chieu rong

; Output:

; - AX: Chu vi

TINH\_CHU\_VI PROC

PUSH BX

ADD BX, CX

MOV AX, 2

MUL BX

POP BX

RET

TINH\_CHU\_VI ENDP

END

XUAT\_THAP\_PHAN PROC

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV BX, 10

XOR CX, CX

XULY\_STACK:

XOR DX, DX

DIV BX

PUSH DX

INC CX

CMP AX, 0

JA XULY\_STACK

PRINT:

POP DX

OR DX, 30h

MOV AH, 2

INT 21h

LOOP PRINT

POP DX

POP CX

POP BX

POP AX

RET

XUAT\_THAP\_PHAN ENDP

1. Nhập vào 2 số, xuất ra tổng hiệu tích thương

; Xuat tich

MOV DX, OFFSET RES3

CALL XUAT\_CHUOI

MOV AX, NUM1

MOV BX, NUM2

MUL BX

CALL XUAT\_THAP\_PHAN

; Xuat thuong

MOV DX, OFFSET RES4

CALL XUAT\_CHUOI

XOR DX, DX

MOV AX, NUM1

MOV BX, NUM2

DIV BX

CALL XUAT\_THAP\_PHAN

MOV AH, 4Ch

INT 21h

; =====================================

; PHAN KHAI BAO THU TUC

; =====================================

XUAT\_CHUOI PROC

PUSH AX

MOV AH, 9

INT 21h

POP AX

RET

XUAT\_CHUOI ENDP

.MODEL SMALL

.STACK 100h

.DATA

MSG1 DB 10,13, 'Nhap vao so thu nhat: $'

MSG2 DB 10,13, 'Nhap vao so thu hai : $'

RES1 DB 10,13, 'Tong: $'

RES2 DB 10,13, 'Hieu: $'

RES3 DB 10,13, 'Tich: $'

RES4 DB 10,13, 'Thuong: $'

NUM1 DW ?

NUM2 DW ?

.CODE

MOV AX, @DATA

MOV DS, AX

; Nhap so thu nhat

INPUT\_1:

MOV DX, OFFSET MSG1

CALL XUAT\_CHUOI

CALL NHAP\_THAP\_PHAN

CMP DX, 0

JE INPUT\_1

MOV NUM1, BX

; Nhap so thu hai

INPUT\_2:

MOV DX, OFFSET MSG2

CALL XUAT\_CHUOI

CALL NHAP\_THAP\_PHAN

CMP DX, 0

JE INPUT\_2

MOV NUM2, BX

; Xuat tong

MOV DX, OFFSET RES1

CALL XUAT\_CHUOI

MOV AX, NUM1

MOV BX, NUM2

ADD AX, BX

CALL XUAT\_THAP\_PHAN

; Xuat hieu

MOV DX, OFFSET RES2

CALL XUAT\_CHUOI

MOV AX, NUM1

MOV BX, NUM2

SUB AX, BX

CALL XUAT\_THAP\_PHAN

XUAT\_THAP\_PHAN PROC

PUSH AX

PUSH BX

PUSH CX

PUSH DX

MOV BX, 10

XOR CX, CX

XULY\_STACK:

XOR DX, DX

DIV BX

PUSH DX

INC CX

CMP AX, 0

JA XULY\_STACK

PRINT:

POP DX

OR DX, 30h

MOV AH, 2

INT 21h

LOOP PRINT

POP DX

POP CX

POP BX

POP AX

RET

XUAT\_THAP\_PHAN ENDP

END

NHAP\_THAP\_PHAN PROC

PUSH AX

PUSH CX

PUSH SI

XOR BX, BX

INPUT:

MOV AH, 1

INT 21h

CMP AL, 0Dh

JE BREAK

CMP AL, '0'

JB XULY\_LOI

CMP AL, '9'

JA XULY\_LOI

AND AL, 0Fh

MOV CL, AL

XOR CH, CH

MOV AX, BX

MOV SI, 10

MUL SI

MOV BX, AX

ADD BX, CX

JMP INPUT

XULY\_LOI:

XOR BX, BX

MOV DX, 0

JMP EXIT

BREAK:

MOV DX, 1

EXIT:

POP SI

POP CX

POP AX

RET

NHAP\_THAP\_PHAN ENDP