第5章 汇编语言程序设计

> 课件来源：以 ch5_v2023（132 页）为骨架；ch5_v2022（125 页）经逐页比对，其内容全部包含于主版本（主版本另多出 TABLE 版冒泡排序算法与代码 p80-83、第4章答案页 p117、第5章答案页 p126-127），无独有内容，故不单独引用。本章是全书编程大题的唯一出处：伪指令/数据定义、运算符求值、DOS 功能调用、顺序/分支/循环/子程序四大结构、BCD 与 ASCII 转换编程几乎必考，期末编程题与程序阅读题基本都从本章例题与习题变形而来。

0. 本章考点清单

考点	常见题型	重要度
循环程序设计（计数法/条件控制法、数组求和/最值/统计）	编程、程序阅读	★★★
数据定义伪指令 DB/DW/DD/DUP 与内存布局	填空、画图、计算	★★★
DOS 系统功能调用（1/2/9/0AH/4CH/6 号）	编程、填空	★★★
分支程序设计（CMP+条件转移、有符号/无符号区分）	编程、程序阅读	★★★
BCD 码与 ASCII 码相互转换（DAA/AAA/AAM 调整）	编程、程序阅读	★★★
冒泡排序（双重循环+交换标志）	编程、程序阅读	★★★
分析运算符 OFFSET/SEG/TYPE/LENGTH/SIZE	填空、计算	★★☆
子程序设计（CALL/RET、现场保护、参数传递）	编程、简答	★★☆
程序框架与返回 DOS（ASSUME、段装填、PSP、FAR+RET）	填空、简答	★★☆
变量/标号三属性、常数书写规则	选择、填空	★★☆
伪指令 EQU / = / LABEL / SEGMENT / PROC / END	选择、填空	★★☆
宏定义 MACRO/ENDM 与宏展开	简答、填空	★★☆
PTR/THIS/HIGH/LOW/SHORT 及关系/逻辑运算符	填空、计算	★★☆
查表程序与 XLAT	编程、填空	★☆☆
程序设计步骤与流程图符号	简答	★☆☆
多分支结构（DO-CASE，教材例5-38）	简答	★☆☆

1. 汇编语言概述（课件 p2-3）

- 汇编语言用**助记符**表示指令的操作码和操作数，是**面向机器**的语言，不同机器有不同的汇编语言（课件 p2）。
- 特点：执行速度快、占用存储空间小、实时性能好；大量用于系统程序、实时控制程序（课件 p2）。
- 三个名词（课件 p3）：
- **汇编语言源程序**：用汇编语言编写的程序；
- **汇编**：把源程序翻译成机器代码目标程序的过程；
- **汇编程序**：自动完成汇编过程的软件（是软件，不是"编写的程序"）。
- 易错警示：「汇编程序」指翻译工具（如 MASM），「汇编语言程序」指用户写的源程序，二者不可混用。

2. 常数、变量和标号（5.1.1，课件 p4-10）

2.1 常数（课件 p4-6）

常数是没有属性的纯数值，汇编期间值已确定，运行中不变。五种书写形式：

类型	后缀	例
二进制	B	00101101B
八进制	O 或 Q	255Q
十进制	D 可省略	1234D、1234
十六进制	H	56H、0B3FH
字符串	引号括起	'A'（=41H）、'B'（=42H）

- 十六进制**必须以数字开头**：凡以 A~F 开头的十六进制数前必须冠 0，如 0B3FH、0FFH（课件 p6）。
- 字符串常数以 ASCII 码存内存，故串常量与整数常量可交替使用（课件 p5）。
- 常数主要以**立即数、位移量**形式出现在指令语句或数据定义伪指令中（课件 p6）。

2.2 变量及其三属性（课件 p7-8）

变量是存放在存储单元中、运行期间可修改的数据。三属性：

属性	含义
段属性 SEGMENT	变量所在段的段首地址
偏移属性 OFFSET	变量的偏移地址
类型属性 TYPE	变量占用存储单元的字节数（字节/字/双字/四字/十字节）

- 变量用 DB/DW/DD 等定义；变量名可选，若使用则代表**数据区第一个数据单元的地址**（课件 p8）。

- 例（课件 p8）：`DB 12H,34H`；`DW 1234H`；`DD 12345678H`；`DB '9BD'` → 内存 39H、42H、44H。

2.3 标号及其三属性（课件 p9-10）

标号是一条指令所在地址的符号，常用作转移/调用指令的目标地址。命名：字母数字串、**首字符必须为字母**、最多 31 个字符、可用下划线（课件 p9）。

属性	含义
段属性	标号所在段的段基址
偏移属性	标号的偏移地址
距离属性（TYPE）	NEAR：只能段内转移/调用；FAR：可被其他段转移/调用

- 易错警示：变量的 TYPE 是字节数（1/2/4...），标号的 TYPE 是距离（NEAR=-1、FAR=-2），两者含义不同（课件 p10、p15）。

3. 表达式与运算符（5.1.2，课件 p11-22）

表达式分两类：**地址表达式**（结果是存储器地址，存数据称变量、存指令称标号）与**数值表达式**（只产生数值）（课件 p11）。运算符六类：

3.1 算术运算符（课件 p12）

`+、-、*、/、MOD(求余)、SHL(左移)、SHR(右移)`。例：`CONT = 14*4`；`MOV AL, 21H SHL 2`（汇编期计算，AL←84H）。

3.2 逻辑运算符（课件 p12）

`AND、OR、XOR、NOT`，**只适用于常数**的逻辑运算（汇编期求值）。例：`AND DX, PORT AND 0FEH`——后一个 AND 是运算符（把 PORT 最低位清 0 后的常数作立即数），前一个 AND 是指令。

3.3 关系运算符（课件 p13）

`EQ、NE、LT、GT、LE、GE`。结果为假→0；为真→**0FFH 或 0FFFFH**（不是 1！）。 例：`MOV BX, ((PORT LT 5) AND 20) OR ((PORT GE 5) AND 30)`：PORT<5 时等价 `MOV BX,20`；PORT≥5 时等价 `MOV BX,30`（课件原文写"PORT>5"，按 GE 含义应含等于，课件原文如此）。

3.4 分析操作符（数值返回运算符，课件 p14-17）

运算符	格式	功能
OFFSET	OFFSET 变量/标号	取段内偏移量
SEG	SEG 变量/标号	取段首地址
TYPE	TYPE 变量/标号	变量返回字节数；标号返回距离值
LENGTH	LENGTH 变量	数据项个数：DUP 定义返回 DUP 前的 n，否则返回 1
SIZE	SIZE 变量	字节数 = LENGTH × TYPE

TYPE 返回值表（课件 p15）：

对象	属性	类型值
变量	BYTE	1
变量	WORD	2
变量	DWORD	4
标号	NEAR	-1（0FFFFH）
标号	FAR	-2（0FFFEH）

例（课件 p17）：`BUF DW 100 DUP(0)`、`ARRAY DW 1,2,3`、`BUF1 DB 100 DUP(0)`、`BUF2 DB 200 DUP(20H)`、`BUF3 DD 100 DUP(13H)`： LENGTH BUF=100、LENGTH ARRAY=**1**、LENGTH BUF1=100、LENGTH BUF2=200、LENGTH BUF3=100； SIZE BUF=200、SIZE ARRAY=**2**、SIZE BUF1=100、SIZE BUF2=200、SIZE BUF3=400。 `MOV CX, SIZE BUF` 等价 `MOV CX, 200`。

3.5 综合运算符（课件 p18-20）

- **PTR**：`类型 PTR 表达式`，临时指定类型，**只在当前指令内有效**。类型可取 BYTE/WORD/DWORD/FAR/NEAR。例：`MOV BYTE PTR [DI],4`；`JMP DWORD PTR [BP]`。又：`TWO_BYTE DW ?` 配 `ONE_BYTE EQU BYTE PTR TWO_BYTE` → 同段同偏移、类型不同（课件 p18）。
- **THIS**：`THIS 类型`，把类型赋给**当前**（下一个）变量/标号，**不分配新存储单元**。例：`A EQU THIS BYTE` 后接 `B DW 20 DUP(?)` → A 按字节、B 按字访问同一地址；`WBUFFER1 EQU THIS WORD` 后接 `BUFFER DB 100 DUP(0)` → 字节表改按字使用（课件 p19-20）。

3.6 其他运算符（课件 p21-22）

- **HIGH / LOW**：分离高/低字节。`K1 EQU 1234H`：`MOV AL,LOW K1`→AL=34H；`MOV BL,HIGH K1`→BL=12H。
- **SHORT**：`SHORT 标号`，说明段内短转移，目标与本指令下一条指令的距离在 **-128~127** 字节内。例：`L1: JMP SHORT L2`。

- 运算符优先级：课件只指引教材 P108 表4-2（课件 p22）。（补充常识）一般顺序：括号 → HIGH/LOW → PTR/THIS/SEG/OFFSET/TYPE/LENGTH/SIZE → */、MOD、SHL、SHR → +、- → 关系运算符 → 逻辑运算符。

- 易错警示：LENGTH 只看**最外层 DUP** 的 n；`DW 1,2,3` 的 LENGTH=1、SIZE=LENGTH×TYPE=2（课件 p17），而不是实际总字节数 6——SIZE 也只按最外层 DUP 计算，这是高频陷阱。

4. 伪指令（5.2，课件 p23-43）

4.1 两类语句与伪指令格式（课件 p23）

- **指令语句**：汇编时翻译成目标代码，连接后成为可执行机器指令。
- **伪指令语句**（指示性语句）：汇编时被解释执行，由汇编程序处理（数据定义、分配存储器等），**不产生机器代码**。
- 格式：`符号名 定义符 操作数 ;注释`。符号名可以是常量名、变量名、段名、过程名；指令语句中标号后加「:」，伪指令的符号名后**不加冒号**；注释以分号开始。

4.2 符号定义伪指令（课件 p24-26）

伪指令	格式	要点
EQU	符号名 EQU 表达式	一旦定义**不能再赋值**
=	符号名 = 表达式	可**多次重新赋值**
LABEL	符号名 LABEL 类型	定义标号/变量并指定类型，与下一语句同地址

例：`ALFA EQU 100`、`COUNT EQU ALFA`；`ALFA = 100` 后 `ALFA = ALFA+2` → 102；`BARY LABEL WORD` 后 `ADD AX,BARY` 为字加法；`ABCF LABEL FAR` 定义 FAR 标号。

4.3 数据定义伪指令（课件 p27-33）

格式：`[变量名] DB/DW/DD/DQ/DT 表达式`，分别定义 1/2/4/8/10 字节（课件 p27）。表达式四种形式：

1. **数值表达式**：逗号分隔依次存放；字数据**低字节存低地址**（课件 p28）。

(1) 数值表达式

实际上是为数据分配存储单元，并把变量名作为该存储单元的名称。若要定义多个相同类型的数据，可用逗号把这些数据隔开，并依次分配在相邻的存储单元。

例：
```
BUF  DB  20H，30H
BUF1 DW 100
DAT DW 1234H，5678H ；
```

字变量DAT的内容为1234H，占用两个字节单元，下一个字单元即DAT+2单元的内容为5678H，仍是低字节存在低地址单元。



28

2. **ASCII 字符串**：引号括起、不超过 256 字符，按地址递增自左向右存 ASCII 码。**DB 'AB' → 41H,42H; DW 'AB' → 42H,41H**（字常量 'AB'=4142H，低字节 42H 在低地址）（课件 p29）。

(2) ASCII字符串

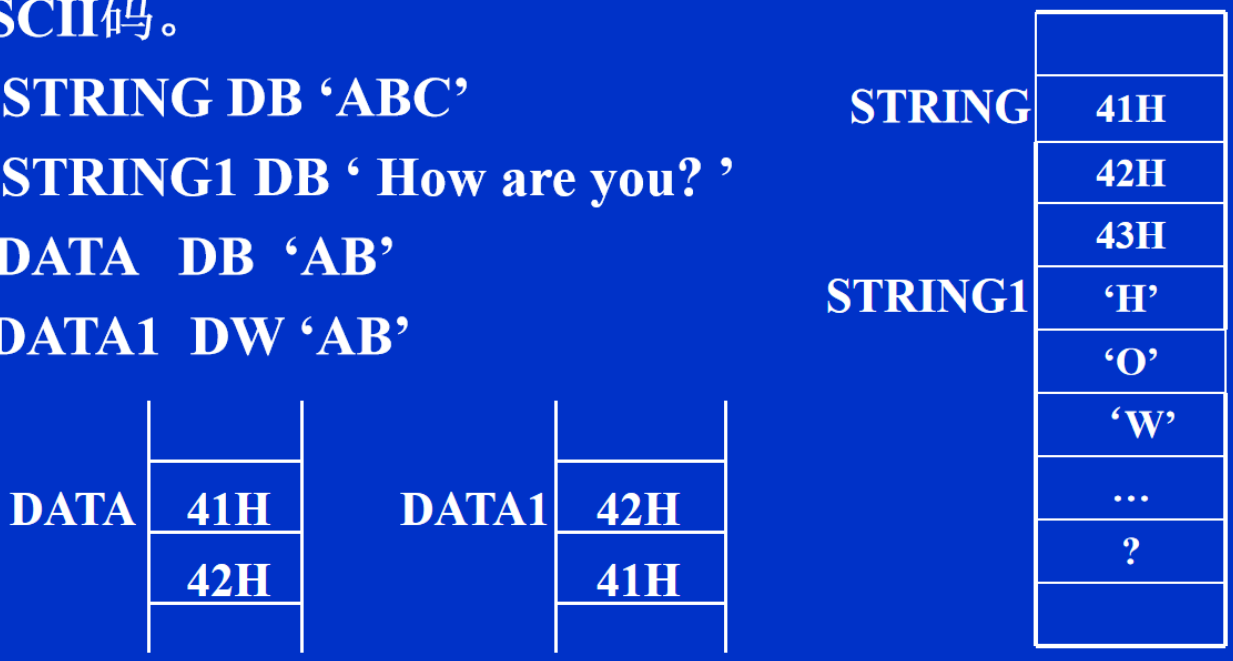
字符串必须用引号括起来，并且不超过256个字符，它们在内存中按地址递增顺序自左向右依次存放字符的ASCII码。

例：STRING DB ‘ABC’

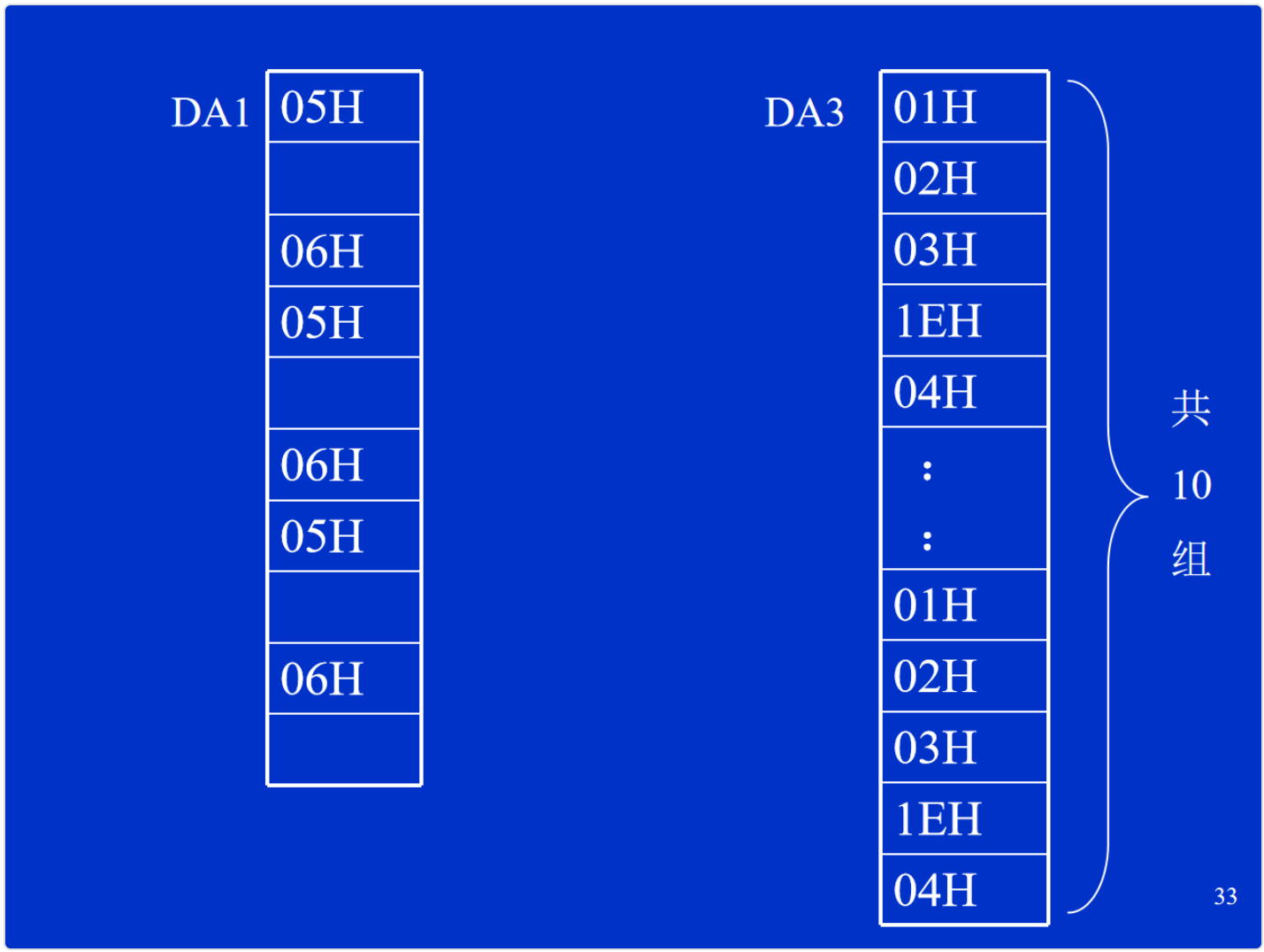
STRING1 DB ‘ How are you? ’

例：DATA DB ‘AB’

DATA1 DW ‘AB’



3. **?**：只分配单元不预置值，用于预留中间/最终结果。例：`ABC DB 12H,?,?,24H`；`DEF DW ?,895H,?`。注意：定义**多于两个字符**的字符串只能用 DB，不能用 DW/DD（课件 p30）。
4. **DUP 重复子句**：`n DUP(表达式)`，可嵌套；同一条数据定义语句定义的单元类型相同（课件 p31）。 - `DA1 DB 3 DUP(5,?,6)` → 9 字节：05H,?,06H 三组； - `DA2 DW 40 DUP(?)` → 80 字节； - `DA3 DB 10 DUP(1,2,3 DUP(30,4))`：课件文字称"十组 01,02,30,04,30,4,30,4，共占用50个字节"（课件 p32）。**存疑**：按所写表达式每组应为 8 字节、共 80 字节；p33 配图与"50字节"对应的是每组 5 字节（01H,02H,03H,1EH,04H），即 `10 DUP(1,2,3,30,4)`。表达式与图/字节数自相矛盾，考试以教材原题为准，记住"嵌套 DUP 逐层展开相乘"的方法即可。



4.4 段定义伪指令（课件 p34-37）

格式：`段名 SEGMENT [定位类型][组合类型]['类别名'] … 段名 ENDS`，首尾段名必须相同。

- **定位类型**（课件 p35）：PAGE=256 字节边界（起始地址被 256 整除）；PARA=16 字节边界（被 16 整除），**缺省默认 PARA**；WORD=偶地址边界；BYTE=任意地址。
- **组合类型**：多模块连接时同名段的组合关系（连接/重叠），课件注明"见书P126，了解内容"（课件 p36）。
- **类别**：单引号括起的自定义符，如 'STACK'、'CODE'；同类别段被依次放在连续存储区但各段仍独立（课件 p37）。

4.5 段寄存器说明 ASSUME（课件 p38-39）

`ASSUME 段寄存器:段名[,…]`。两点必考： 1. 只是**告诉汇编程序**对应关系，**并不装填段寄存器**；除 CS 外，DS/ES/SS 的实际值必须用 MOV 指令装填（如 `MOV AX,DATA` / `MOV DS,AX`）。 2. 可用 `ASSUME DS:NOTHING` 取消对应关系后再建立新的。

4.6 过程定义 PROC/ENDP（课件 p40-41）

`过程名 PROC [NEAR/FAR] … RET … 过程名 ENDP`。过程名首尾相同，有段地址、偏移地址、距离三属性；可有多个 RET，一般作为最后一条指令。例：`APRC PROC NEAR … RET … APRC ENDP`，主程序 `CALL APRC`，`END STAR` 指定入口。

4.7 模块定义 NAME / END（课件 p42-43）

- `NAME 模块名`：模块开始；缺省时取 TITLE 页标题前 6 个字符，无 TITLE 则取源文件名。
- `END 启动标号`：模块结束并给出**程序入口**；汇编处理只进行到 END 为止。

5. 宏指令（5.3，课件 p44-46）

- 宏定义：`宏指令名 MACRO [形参,…] … ENDM`；宏调用：`宏指令名 [实参,…]`（课件 p44）。
- 例：左移 4 位序列 `MOV CL,4 / SAL AL,CL` 多次使用时定义为宏 SHIFT（课件 p45）：

```
SHIFT   MACRO
        MOV  CL,4
        SAL  AL,CL
        ENDM
; 调用处：
        MOV  AL,[DI]
        ADD  AL,[SI]
        SHIFT            ; 宏调用，汇编时展开为两条指令
        MOV  [BX],AL
```

- 带参数的宏（课件 p46）：`SHIFT MACRO X / MOV CL,X / SAL AL,CL / ENDM`，调用 `SHIFT 4` 时形参 X 被实参 4 替换。
- 要点（补充常识，与课件宏展开例子一致）：宏在**汇编时展开**，每次调用都插入一份宏体代码，无调用/返回开销但增大代码量；子程序在**运行时** CALL/RET，代码只有一份。宏适合短小、频繁使用的代码段。

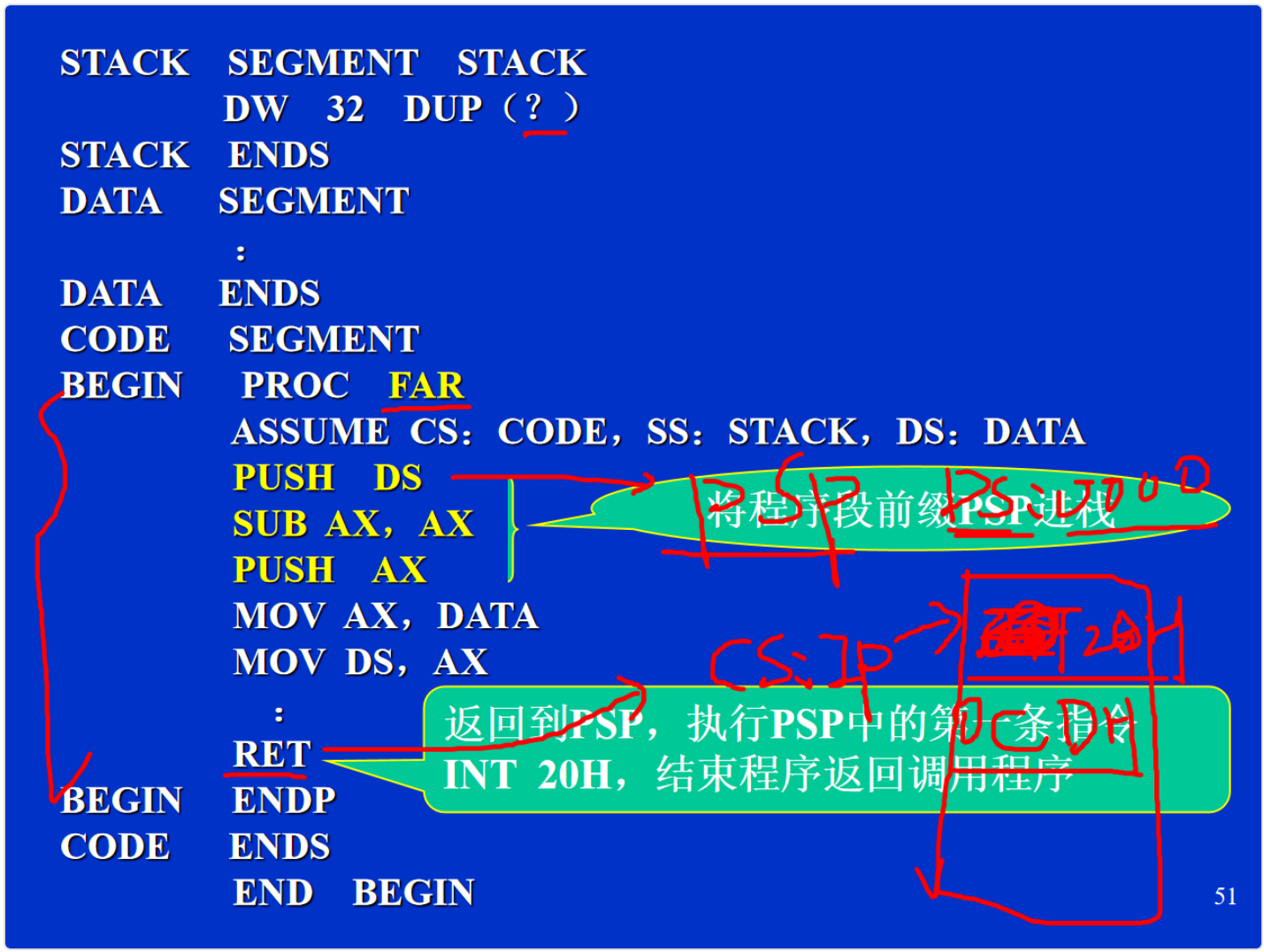
6. 汇编语言程序的结构（5.4，课件 p47-52）

6.1 标准源程序框架（课件 p47-48）

```
DATA    SEGMENT                  ; 定义数据段
        ...
DATA    ENDS
STACK   SEGMENT PARA STACK 'STACK'   ; 定义堆栈段及长度
        ...
STACK   ENDS
CODE    SEGMENT                  ; 定义代码段
        ASSUME CS:CODE,SS:STACK,DS:DATA,ES:DATA
START:  MOV  AX,DATA             ; 装填用户数据段、附加段
        MOV  DS,AX
        MOV  ES,AX
        MOV  AX,STACK            ; 装填用户堆栈段
        MOV  SS,AX
        ...
        MOV  AH,4CH              ; 返回DOS
        INT  21H
PRA     PROC NEAR
        ...
PRA     ENDP
CODE    ENDS
        END  START
```

6.2 与 DOS 的接口：PSP（课件 p49-51）

- 连接程序为每个用户程序建立**程序段前缀区 PSP**，长 **256 字节**，存放程序信息及与操作系统的接口；PSP 段地址在 **DS 和 ES** 中，偏移 0 处是一条 **INT 20H**（课件 p49）。
- 传统返回法两措施（课件 p50）：① 主程序定义为 **FAR 过程**，最后一条指令为 RET；② 程序开始处把 PSP 段基址 DS 压栈，再压入一个全 0 字（构造远返回地址 DS:0000）。




```
CODE    SEGMENT
BEGIN   PROC FAR
        ASSUME CS:CODE,SS:STACK,DS:DATA
        PUSH DS              ; PSP 段基址进栈
        SUB  AX,AX
        PUSH AX              ; 全0偏移进栈 → 栈中形成返回地址 DS:0000
        MOV  AX,DATA
        MOV  DS,AX
        ...
        RET                  ; 远返回到 PSP:0000，执行 INT 20H 结束程序
BEGIN   ENDP
CODE    ENDS
        END  BEGIN
```

6.3 返回 DOS 的两种方法（课件 p52）

1. `MOV AH,4CH / INT 21H`（4CH 号功能调用，现代写法，推荐）；

2. `INT 20H`（机器码 CD20H；DEBUG 加载 EXE 后在 DS:0000 可见），要求 CS 指向 PSP 或按 6.2 构造返回地址。

7. DOS 系统功能调用（5.5，课件 p53-60）

- DOS 常用软中断 8 条，类型码 **20H~27H**；**INT 21H** 是 DOS 系统功能调用，内含几十个子程序（课件 p53）。
- 统一调用格式三步：① 入口参数送指定寄存器；② **功能号送 AH**；③ `INT 21H`（课件 p53）。

功能号	功能	入口参数	出口/说明
01H	键盘输入单字符	无	AL=字符 ASCII，回显；先检查 Ctrl-Break（课件 p54）
02H	输出单字符	DL=字符	送屏幕显示（课件 p55）
06H	直接控制台 I/O	DL=0FFH 输入；DL≠0FFH 输出	输入时字符→AL（课件 p58）
0AH	键盘输入字符串	DS:DX=缓冲区首址	缓冲区三段结构，见下（课件 p56）
09H	输出字符串	DS:DX=串首址	串以 **'$'** 结束（课件 p57）
4CH	返回操作系统	无	结束程序（课件 p58）

0AH 号缓冲区结构（课件 p56）：第 1 字节=缓冲区容量；第 2 字节=预留，系统回填**实际输入字符数**；第 3 字节起存放字符串，以**回车 0DH** 结束。

```
BUF     DB   20              ; 容量
        DB   ?               ; 实际长度，系统填写
        DB   20 DUP(?)       ; 字符串存放区
        ...
        MOV  DX,OFFSET BUF
        MOV  AH,0AH
        INT  21H
```

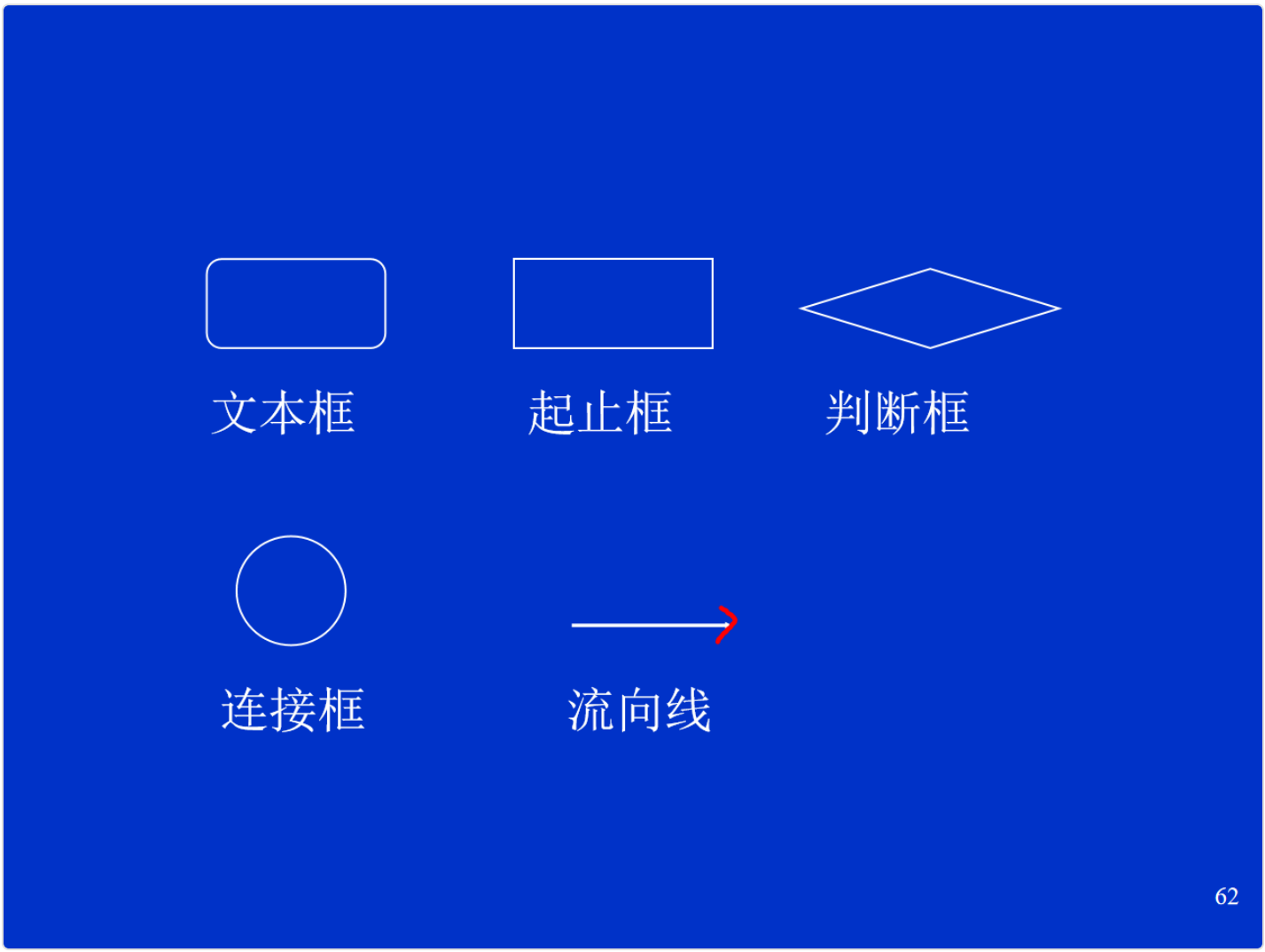
例5-35 人机对话（课件 p59-60）：9 号显示提问串 `MSG DB 'WHAT IS YOU NAME? $'`，0AH 号接收姓名到 BUF，4CH 返回。完整代码见习题册题1。

- 易错警示：9 号串结束符是 '$'（24H），0AH 输入结束符是 0DH，二者不同；0AH 的第 2 字节不是用户填的，是系统回填的实际长度。

8. 程序设计步骤与流程图（5.6.1，课件 p61-62）

七步（课件 p61）：① 分析问题 → ② 建立数学模型 → ③ 确定算法 → ④ 绘制程序流程图 → ⑤ 内存空间分配 → ⑥ 编制程序与静态检查 → ⑦ 程序调试。 编写目标：结构清晰、可读性强、占用存储空间少、运行速度快。

流程图五种符号（课件 p62）：



- 课件标注：圆角矩形=文本框、矩形=起止框、菱形=判断框、圆=连接框、箭头=流向线。**存疑**：与通用惯例（圆角矩形=起止框、矩形=处理/文本框）相反，课件原文如此；考试若考符号名称以课件/教材表述为准，若考画法按惯例理解即可。

9. 顺序程序设计（课件 p63-66）

9.1 多项式求值（秦九韶/霍纳法）

例：$f(X)=5X^3+4X^2-3X+2$，$X=5$。变形 **$f(X)=((5X+4)X-3)X+2$**，把乘法次数降到 3 次（课件 p63）。

```
        MOV  AX,5
        MUL  X            ; 5*X
        ADD  AX,4         ; 5X+4
        MUL  X            ; (5X+4)X
        SUB  AX,3         ; (5X+4)X-3
        MUL  X            ; ((5X+4)X-3)X
        ADD  AX,2         ; +2
        MOV  RESULT,AX    ; X=5 时结果 712 (02C8H)
```

要点：字乘法 `MUL X` 结果在 DX:AX，本题数值小 DX=0 故只取 AX；若 X 较大必须考虑 DX（易错）。

9.2 查表程序与 XLAT（课件 p65-66）

TABLE 起 7 个单元存 0~6 的立方（0,1,8,27,64,125,216），X 在 FIRST 单元，结果送 SECOND。 方法一（地址计算）： `BX=OFFSET TABLE`， `AX=X`（AH 清 0）， `ADD BX,AX`， `MOV AL,[BX]`。 方法二（查表指令）：

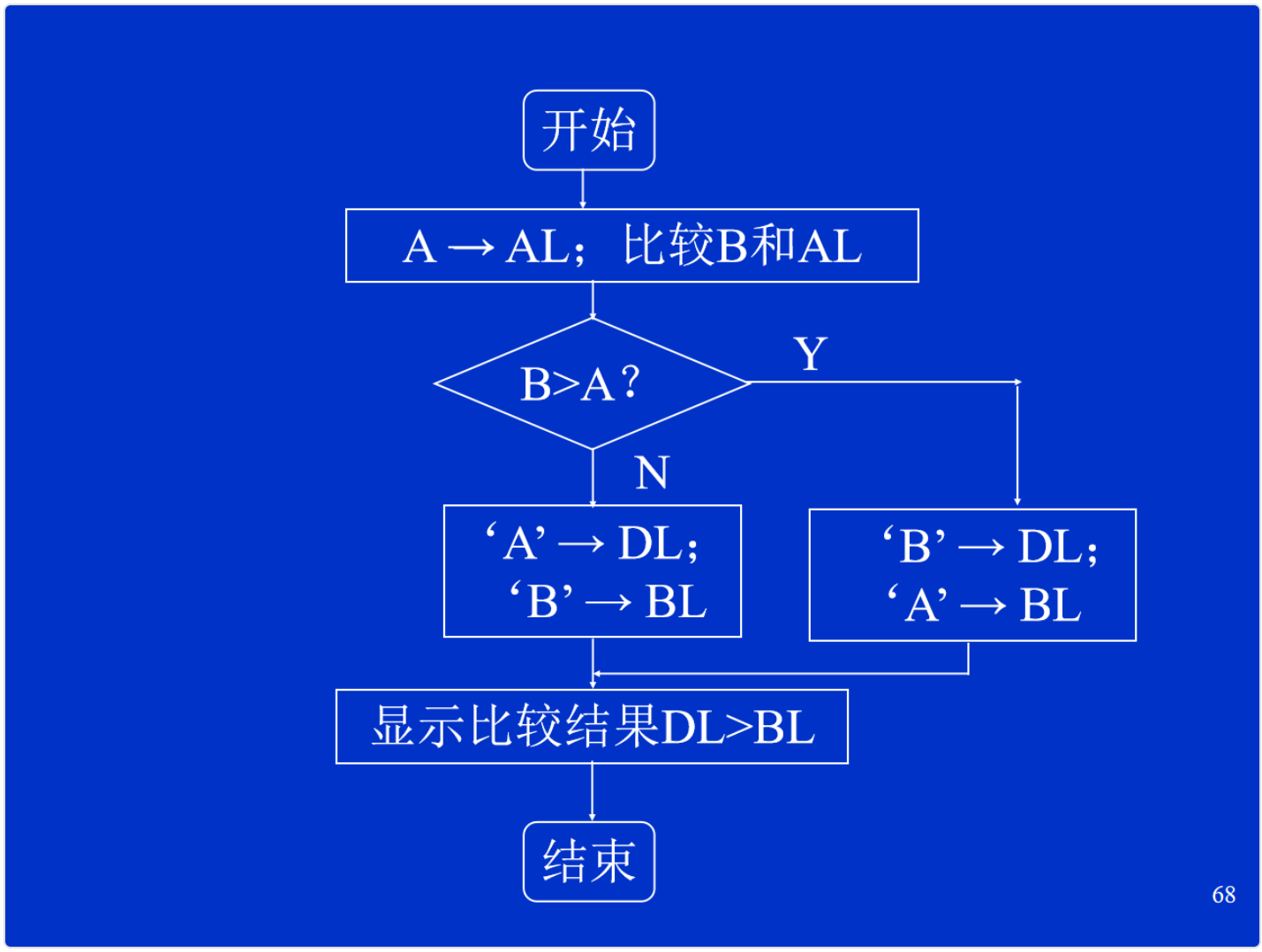
```
        MOV  BX,OFFSET TABLE
        MOV  AL,FIRST
        XLAT              ; AL ← DS:[BX+AL]
        MOV  SECOND,AL
```

- XLAT 功能：以 BX 为表首址、AL 为索引，取 [BX+AL] 单元内容送 AL。课件注释写作"(AL+BX)→AL"，应理解为**取该地址单元的内容**（课件 p66）。
- 注意课件 `FIRST DB X` 中 X 应为 0~6 的常数符号（课件简写，未给出 X 的 EQU 定义）。

10. 分支程序设计（课件 p67-71）

10.1 简单分支（IF-THEN / IF-THEN-ELSE）

例：A、B 单元各一个 8 位**无符号**数（不相等），显示 "A>B" 或 "B>A"（课件 p67）。流程：



关键代码（课件 p69，A=20H、B=92H）：

```
        MOV  AL,A
        CMP  B,AL          ; B - A
        JA   BGA           ; 无符号高于则转（CF=0且ZF=0）
        MOV  DL,'A'        ; A>B：先显示大者
        MOV  BL,'B'
        JMP  DISP
BGA:    MOV  DL,'B'
        MOV  BL,'A'
DISP:   MOV  AH,2
        INT  21H           ; 显示 DL
        MOV  DL,'>'
        INT  21H
        MOV  DL,BL
        INT  21H           ; 显示小者
```

本例运行结果：显示 **B>A**。要点：无符号比较用 JA/JB，有符号用 JG/JL；`CMP B,AL` 是"内存-寄存器"形式，方向决定转移条件写法。

10.2 多分支（DO-CASE）

课件 p70 仅指出多分支相当于高级语言 DO-CASE 结构，具体见教材 P145 例5-38（课件未给代码）。（补充常识）常用**转移地址表法**：把各分支入口偏移地址存入 DW 表，用索引×2 查表后 `JMP` 间接转移。

10.3 分支+循环求最值（课件 p71）

求 10 个**带符号**数最大者：`CX=9`（n-1 次比较），AL 先取首数，循环内 `CMP AL,[SI+1] / JG NEXT / MOV AL,[SI+1]`，`INC SI / DEC CX / JNZ`。有符号比较用 **JG**；比较次数 = 个数-1。完整代码与逐行注释见习题册题5。

11. 循环程序设计（课件 p72-85）

11.1 循环四部分（课件 p72）

① **初始化**：地址指针、计数器、参数初值；② **循环体**（工作部分）：实际处理；③ **循环控制**：修改指针/计数器、判断结束条件；④ **结束部分**：分析和存放结果。相当于 DO-UNTIL / DO-WHILE。

11.2 循环控制方法（课件 p73）

- **计数法**：已知次数，常用 **CX 作计数器**（LOOP 指令自动 DEC CX 并判非零）。
- **条件控制法**：次数未知，靠条件（如串结束符）退出，用 CMP+JZ/JNZ 等。

11.3 三个基本例（课件 p74-79）

- 输出 0~9（计数法）：CX=0AH，BL=30H 起，2 号功能显示，INC BL、DEC CX、JNZ（课件 p74）。
- 统计负数个数并显示（计数法）：20 个 8 位带符号数，`AND AL,AL` 置标志，`JNS NEXT` 跳过正数，负数 `INC DL`；LOOP 控制；结果存 NEGA 并 `OR DL,30H` 转 ASCII 显示（设个数 0~9）（课件 p75-77）。
- 统计字符串长度（条件控制法）：以 '$' 结束，`CMP AL,[DI] / JZ STOP`，否则 `INC LEN / INC DI / JMP LP`（课件 p78-79）。

11.4 冒泡排序（双重循环，课件 p80-85）

版本一（TABLE，6 个数，课件 p80-83） 算法十步（课件 p80）：外循环次数→CH；表首址→SI；内循环次数→CL；交换标志 BL 清 0；相邻比较顺序对则跳过；不对则交换并置 BL=1；SI+1；CL-1 非零继续内循环；BL=0 说明已排好→结束；否则 CH-1 非零继续外循环。

```
DATA    SEGMENT
TABLE   DB  12,78,-2,0,75,15
LEN     EQU $-TABLE            ; =6
DATA    ENDS
CODE    SEGMENT
        ASSUME CS:CODE,DS:DATA
START:  MOV  AX,DATA
        MOV  DS,AX
        MOV  CH,LEN
        DEC  CH                ; 外循环次数 = n-1 = 5
LOP0:   MOV  SI,OFFSET TABLE
        MOV  CL,CH             ; 内循环次数逐趟递减
        XOR  BL,BL             ; 交换标志清0
LOP1:   MOV  AL,[SI]
        CMP  AL,[SI+1]
        JLE  NEXT              ; 带符号：顺序正确跳过
        XCHG AL,[SI+1]         ; 交换两相邻数
        MOV  [SI],AL
        OR   BL,01H            ; 置交换标志
NEXT:   INC  SI
        DEC  CL
        JNZ  LOP1
        AND  BL,BL
        JZ   STOP              ; 本趟无交换 → 已有序，提前结束
        DEC  CH
        JNZ  LOP0
STOP:   MOV  AH,4CH
        INT  21H
CODE    ENDS
        END  START
```

版本二（M1，100 个带符号数，从后往前扫+边界收缩，课件 p84-85）：

```
        MOV  BL,02;外循环初值
NEXT3:MOV SI,OFFSET M1+99
        MOV  CH,64H;设交换标志
        MOV  CL,64H ；  内循环初值
NEXT2:MOV AL,[SI]
        CMP  AL,[SI-1]
        JGE  NEXT1
        XCHG  AL,[SI-1]
        MOV [SI],AL
        MOV CH,CL
NEXT1:DEC  SI
        DEC  CL
        CMP  CL,BL
        JGE NEXT2
        CMP CH,64H
        JZ  DONE
        INC  BL
        JMP  NEXT3
DONE:MOV AH,4CH
        INT 21H
CODE  ENDS
        END  START
```

```
        MOV  BL,02              ; 内循环下界初值
NEXT3:  MOV  SI,OFFSET M1+99    ; 从表尾开始
        MOV  CH,64H             ; 交换标志=100（未交换）
        MOV  CL,64H             ; 内循环计数初值100
NEXT2:  MOV  AL,[SI]
        CMP  AL,[SI-1]
        JGE  NEXT1              ; 带符号：[SI]>=[SI-1] 顺序正确
        XCHG AL,[SI-1]
        MOV  [SI],AL
        MOV  CH,CL              ; 记录发生交换（标志≠64H）
NEXT1:  DEC  SI
        DEC  CL
        CMP  CL,BL
        JGE  NEXT2              ; 内循环：CL>=BL 继续
        CMP  CH,64H
        JZ   DONE               ; 本趟无交换 → 结束
        INC  BL                 ; 下界上移1（最小者已沉底）
        JMP  NEXT3
DONE:   MOV  AH,4CH
        INT  21H
```

- 两版本对比：版本一从前往后、每趟把最大者沉到表尾，外循环 CH 递减；版本二从后往前、每趟把最小者浮到表头，下界 BL 递增；都用交换标志实现"已有序提前退出"。
- 易错警示：带符号数排序必须用 JLE/JGE（版本一 JLE、版本二 JGE），误用 JBE/JAE 会把负数当大数。

12. 子程序设计（课件 p86-89）

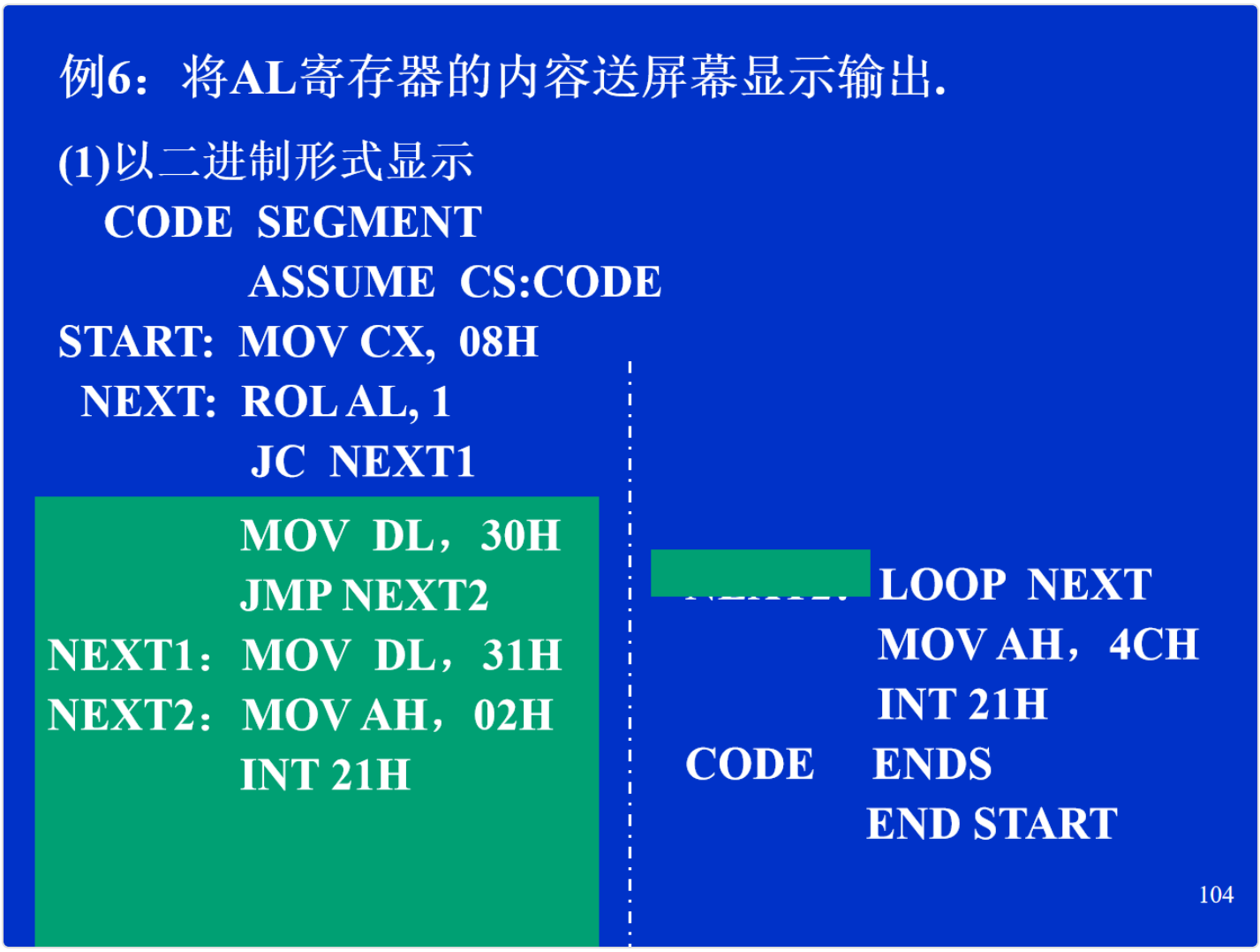
- 具有公用性、重复性、相对独立性的程序段应设计成子程序（课件 p86）。
- 调用与返回由 **CALL / RET** 实现；调用即转移但**要保护返回地址**（CALL 压栈 IP 或 CS:IP）。
- 设计三注意（课件 p86）：① **现场保护和恢复**（PUSH/POP 成对）；② **参数传递**：寄存器传递、内存单元传递；③ **子程序说明**（功能、入口/出口参数、占用寄存器）。
- 例5-42（课件 p87-89）：LIST DB 16,48,?；ADSUB 子程序求和存 LIST+2；OUTPR 子程序用 `DIV BL`(BL=10) 分离十位/个位（商 AL、余数 AH），两次调用 DISPL（OR DL,30H 后 2 号显示）输出 "64"。**存疑**：课件 DISPL 过程体内**漏写 RET**（p88），若照抄会顺序落入 OUTPR 造成递归死循环；正确写法应在 INT 21H 后加 RET（见习题册题11）。

13. 综合程序设计举例：BCD 与 ASCII 转换（5.6.3，课件 p90-108）

本节是编程大题题库，八个例子覆盖全部转换方向。调整指令用法速记：

指令	作用	课件出处
DAA	压缩 BCD 加法调整（AL）	p92
DAS	压缩 BCD 减法调整（AL）	p119(4-6)
AAA	非压缩 BCD 加法调整（AL，进位入 AH）	p93、p96、p100
AAM	非压缩 BCD 乘法调整（AX→AH 十位、AL 个位）	p96

1. **多字节压缩 BCD 加法**（p90-92）：10 字节、低位在低地址；`CLC` 后循环 `MOV AL,[SI] / ADC AL,[DI] / DAA / MOV [BX],AL`，三指针 INC，LOOP；循环外 `MOV AL,0 / ADC AL,0 / MOV [BX],AL` 存最高进位（C 定义 11 DUP(?)）。减法同理改 SBB+DAS。
2. **ASCII 十进制数相加练习**（p93）：先 `AND AL,0FH` 屏蔽高 4 位（35H→05H），再 AAA 调整；课件只给分析与数据段，完整程序见习题册题13。
3. **非压缩十进制乘法**（p94-96）：多位×一位；循环 `MUL B / AAM / ADD AL,[DI] / AAA / 存AL / INC DI / 存AH`；AAM 把乘积分成十位个位，AAA 处理与上一轮高位的相加进位。例 6357×8=50856。
4. **非压缩 BCD→二进制**（p97-98）：按权展开累加，`MUL M2`(×10)、`MUL M1`(×100)，结果累加进 DX；例 2,5,6 → 652=028CH。
5. **键盘两个一位十进制数相加显示**（p99-100）：1 号输入两次，ASCII 直接相加后 `MOV AH,0 / AAA`，`OR AX,3030H` 一次转两个 ASCII，先显 BH（十位）再显 BL（个位）。
6. **压缩 BCD→ASCII 并显示**（p101-103）：每字节拆两半；高半 `SHR AL,CL`(CL=4) 后 OR 30H 显示，低半 `AND DH,0FH` 后 OR 30H 显示，字节间显空格；**PUSH CX/POP CX 保护循环计数**（CL 被移位占用）。
7. **AL 内容二进制/十六进制显示**（p104-106）： - 二进制：CX=8，`ROL AL,1` 把最高位旋入 CF，`JC` 分支送 '1'/'0'，2 号显示，LOOP；8 次后 AL 复原。



- 十六进制（分支版 p105 / 循环版 p106）：先 `SHR BL,CL` (4) 取高半字节，`CMP BL,9 / JA` 则 `ADD BL,37H` 否则 `OR BL,30H`；低半 `AND BH,0FH` 同法。循环版用 CH=2 控制两遍。转换规则：**0~9 加 30H，A~F 加 37H**。 8. **ASCII→压缩 BCD（原地压缩）**（p107-108）：100 个数字 ASCII → 50 字节；CH=32H(50) 计数，`AL=[SI] AND 0FH`（低半），`BL=[SI+1] SHL 4`（高半），OR 后存 [DI]；SI 步长 2、DI 步长 1，读指针始终超前写指针故原地安全。

- 易错警示：AAA/AAM 只调整 AL（进位/十位进 AH），不能用于压缩 BCD；压缩 BCD 加减必须 DAA/DAS 且配合 ADC/SBB 传递进位。

14. 易错点与高频陷阱汇总

1. 以 A~F 开头的十六进制数必须前冠 0（0B3FH），否则被当作符号名（课件 p6）。
2. `DW 'AB'` 内存为 42H,41H，`DB 'AB'` 为 41H,42H——字常量低字节在低地址（课件 p29）。
3. LENGTH 只返回最外层 DUP 的 n，非 DUP 定义返回 1；SIZE=LENGTH×TYPE，`DW 1,2,3` 的 SIZE=2 而非 6（课件 p16-17）。
4. 关系运算符为真时结果是 0FFH/0FFFFH，不是 1（课件 p13）。

5. 标号 TYPE：NEAR=-1、FAR=-2；变量 TYPE 才是字节数（课件 p15）。
6. EQU 定义后不可再赋值，"=" 可以重复赋值（课件 p24-25）。
7. ASSUME 不装填段寄存器；DS/ES/SS 必须 MOV 装填，CS 由系统装填（课件 p38）。
8. PTR 临时改类型只在本指令有效；THIS/EQU THIS 不分配新单元（课件 p18-20）。
9. 逻辑运算符 AND/OR/XOR/NOT 作运算符时只用于常数汇编期求值，别与同名指令混淆（课件 p12）。
10. 有符号比较 JG/JL/JGE/JLE，无符号 JA/JB/JAE/JBE；求带符号最值、排序用错必丢分（课件 p71、p82、p85）。
11. n 个数求最值只需 n-1 次比较；课件习题1答案 CX=20 会多比一次、越界读到 MAX 单元（课件 p111，存疑见习题册）。
12. 字数组指针步长为 2：课件 5-12 答案 `WORD PTR [SI+1]` + `INC SI` 是笔误，应为 [SI+2] 且 SI 加 2（课件 p129，存疑）。
13. CX 既作 LOOP 计数又作移位次数时必须 PUSH CX/POP CX 保护（课件 p102）。
14. 0AH 缓冲区第 2 字节由系统回填实际长度；9 号串以 '$' 结束、0AH 输入以 0DH 结束（课件 p56-57）。
15. `DIV BL`：商 AL、余数 AH；字除法被除数在 DX:AX，忘清 DX 会出错（课件 p88、p64）。
16. AAA/AAM 针对非压缩 BCD（AL），DAA/DAS 针对压缩 BCD；混用结果错（课件 p92-96）。
17. 宏在汇编期展开、子程序在运行期调用；宏无 CALL/RET 开销但代码膨胀（课件 p44-46）。
18. FAR 主过程+PUSH DS/PUSH 0+RET 才能回到 PSP 的 INT 20H；直接 NEAR+RET 会跑飞（课件 p49-51）。
19. XLAT 是 AL←[BX+AL]，BX 表首址、AL 索引，表长不超过 256 字节（补充常识）（课件 p66）。
20. 课件数据段中 BCD 字节常以十进制书写（如 `DB 50,40,25` 实指 50H,40H,25H），读题时按 BCD 值理解（课件 p113、p115）。
21. 子程序 DISPL 类显示过程末尾必须有 RET，课件例5-42 漏写（课件 p88，存疑）。
22. 原地转换（ASCII→BCD、BCD→ASCII 同区）必须保证读指针超前写指针或从后往前处理，否则覆盖未读数据（课件 p108、p115-116）。

15. 典型例题精讲

例1 冒泡排序完整跟踪（TABLE 版，课件 p80-83）

数据 12,78,-2,0,75,15（带符号）。CH=5 趟上限。 - 第1趟 CL=5：比较 (12,78) 不换；(78,-2) 换→12,-2,78,0,75,15；(78,0) 换→12,-2,0,78,75,15；(78,75) 换→12,-2,0,75,78,15；(78,15) 换→12,-2,0,75,15,78。BL=1。 - 第2趟 CL=4：(12,-2) 换→-2,12,0,75,15,78；(12,0) 换→-2,0,12,75,15,78；(12,75) 不换；(75,15) 换→-2,0,12,15,75,78。BL=1。 - 第3趟 CL=3：全部顺序正确，BL=0 → `AND BL,BL / JZ STOP` 提前退出。 结果：-2,0,12,15,75,78。教学要点：外循环上限 n-1、内循环逐趟减 1、交换标志提前终止三件套；JLE 保证带符号语义。

例2 多字节压缩 BCD 加法的进位链（课件 p90-92）

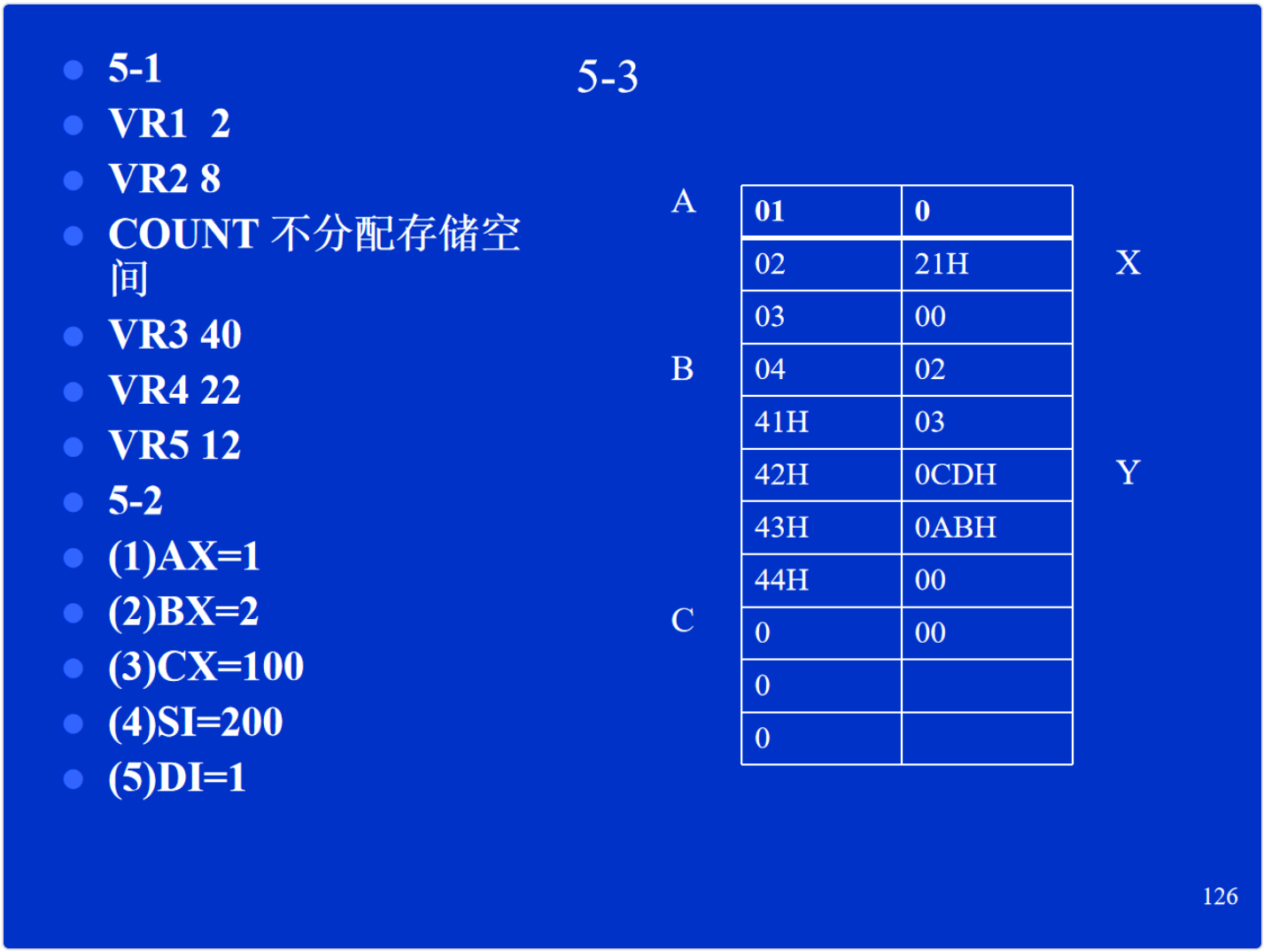
设 A 低字节 12H、B 低字节 88H：`ADC AL,[DI]`（首趟 CF=0）→ 12H+88H=9AH；`DAA`：低半 A>9 加 06H → A0H；高半 A>9 加 60H → 00H 且 CF=1 → 本位 00H、进位 1（12+88=100 ✓）。下一趟 `ADC` 自动带入

CF=1，实现十进制进位链；循环结束后再 `ADC AL,0` 收最高进位存入第 11 字节。教学要点：CLC 只在循环前一次；DAA 必须紧跟 ADC；进位存字节而非丢弃。

例3 十六进制显示的 nibble 拆分（课件 p106 循环版）

设 AL=0B7H。BL=BH=B7H；CH=2、CL=4；SHR BL,4 → BL=0BH。第1遍：CMP 0BH,9 → JA → ADD 37H → 42H='B' 显示；AND BH,0FH → BH=07H；MOV BL,BH；DEC CH=1 → 第2遍：07H≤9 → OR 30H → 37H='7' 显示；DEC CH=0 退出。屏幕输出 "B7"。教学要点：>9 加 37H（=30H+7）而非 30H；循环版用 CH 计数、CL 固定移位次数，避免 CX 冲突。

例4 数据定义内存布局读图（课件 p126，5-3 答案图）



读法：左栏 11 字节接右栏，共 20 字节。A 起：01,02,03,04（A DB 1,2,3,4）；B 起：41H,42H,43H,44H（B DB 'ABCD'）；C 起：4 个 00H；X 起：21H,00H,02H,03H（字变量 DW 21H,302H，低字节在低地址）；Y 起：0CDH,0ABH,00H,00H（DD 0000ABCDH）。教学要点：先按 DB/DW/DD 定宽度，再按"低字节低地址"填字/双字，字符串按 ASCII 顺排；题干见教材，课件仅给答案图。

附：本章速查表

数据定义宽度：DB=1、DW=2、DD=4、DQ=8、DT=10 字节；DUP 嵌套逐层相乘；? 只预留不初始化。

分析运算符：OFFSET=偏移；SEG=段基址；TYPE：BYTE1/WORD2/DWORD4/NEAR-1/FAR-2；LENGTH=最外层 DUP 的 n（否则1）；SIZE=LENGTH×TYPE。

DOS 功能调用（INT 21H）：AH=1 键入回显→AL；AH=2 显示 DL；AH=6 DL=FFH 键入→AL 否则显示 DL；AH=9 显示 DS:DX 串（'$' 结尾）；AH=0AH 键入串到 DS:DX 缓冲区（容量/实际长/内容，0DH 结尾）；AH=4CH 返回 DOS。

ASCII 转换：数字 0~9 ↔ ASCII 加/减 30H；十六进制字符 A~F 加 37H；小写→大写减 20H；压缩 BCD 拆半：高半 SHR 4、低半 AND 0FH；合半：低半 OR (高半 SHL 4)。

BCD 调整：压缩加 DAA、压缩减 DAS、非压缩加 AAA、非压缩乘 AAM；多字节用 ADC/SBB 串进位，循环前 CLC。

比较转移：无符号 JA/JAE/JB/JBE；有符号 JG/JGE/JL/JLE；相等 JE/JZ、不等 JNE/JNZ；LOOP=CX-1 非零转。

程序框架口诀：三段定义（DATA/STACK/CODE）→ ASSUME → 装填 DS/ES/SS → 主逻辑 → 4CH 返回 → ENDS → END 入口。

流程图符号（课件口径）：文本框（圆角矩形）、起止框（矩形）、判断框（菱形）、连接框（圆）、流向线（箭头）；与惯例相反处见 §8 存疑说明。