计算机程序设计基础（1）

05 程序设计结构（上）

清华大学电子工程系

杨昉

E-mail: fangyang@tsinghua.edu.cn

课程回顾

- **C语言表达式**
  - 赋值运算：变量名 = 表达式；**左边必须是变量名**
  - 类型转换：**强制与自动**，**逐步转换**；逻辑运算：**部分成立即结束**
  - 算术运算、关系运算、逻辑运算、单目运算符(++、--)

  单目运算符>！ > 算术运算符 >关系运算符>&&>||>赋值运算>逗号运算符
- **标准函数与宏定义**
  - 标准函数：**库函数**
  - 宏定义：符号常量的宏定义、带参数的宏定义、带#的宏定义

课程回顾:依照优先级和规则进行运算符和运算对象间的计算

类型	运算符	说明
赋值运算符	=、+=、-=、*=、/=、%=	从右到左，自动进行类型转换，还可强制转换
算术运算符	*、/、+、-、%	从左到右,(*,/,%)优先级高,能先执行就先执行，%只适用整型,从低到高且逐步转换，但数据不变
关系运算符	>、>=、<、<=、==、！=	(<,<=,>,>=)优先级高，注意==用法
逻辑运算符	&&、||、！	从左到右，能先执行就先执行，不是所有都执行，！>&&>||,可赋给整型或字符型,非假(0)即真(1)
单目运算符	++、--、(+、-)	在前在后有别,不能有空格，常量与表达式不可用
逗号运算符	,	分隔符或逗号表达式(最后一个子表达式值)
sizeof	sizeof 或 sizeof( )	某种变量或数据类型在计算机所占字节数
位运算符	&、|、^、~、>>、<<	后续会讲

单目运算符 > ！ > 算术运算符 > 关系运算符 > && > || > 赋值运算 > 逗号运算符

课程回顾:按照规则和参数进行无语法检查的字符串替换

类型	定义方法	说明
符号常量定义	#define 符号常量名 字符串	带有符号常量名的地方用字符串进行替换
带参数宏定义	#define 宏名(参数表) 字符串	不仅对字符串进行替换，还进行参数替换。需将参数和字符串都括起来
带#的宏定义(字符串化)	#define 宏名(参数表) #<标识符>	把标识符中参数替换为参数对应字符串
带#的宏定义(字符串连接)	#define 宏名(参数表) <标识符> ## <宏变量>	把标识符中参数的字符串与宏变量字符串连接起来

符号常量名一般用大写字母；只进行简单替换，不做语法检查；双引号字符串不替换；多参数间加逗号；#define不加分号且独占行；可用续行符跨行定义；需注意作用域范围

课程目标

5.1 语句和程序结构

- 程序设计语句
- 基本结构

5.2 顺序结构设计

- 基本概念
- 顺序结构举例

5.3 分支结构设计

- if语句
- 条件运算符
- switch结构

5.1 语句和程序结构

- 程序设计语句
  - ✓ 基本概念
  - ✓ 结构化与非结构化程序
- 基本结构
  - ✓ 顺序结构
  - ✓ 分支结构
  - ✓ 循环结构

语句和程序结构：程序设计语句

- C程序**最小的独立单位是语句(statement)**，语句是组成程序的单元
- 类型
  - **申明语句**，如：int a;
  - **表达式语句**，如：表达式;
  - **执行语句**，如：控制、函数等语句;
  - **空语句**：;
  - **复合语句**：通过{…}来实现的语句组合，如：{int a=3; b=a;}

语句和程序结构：程序设计语句

- 复合语句
  - 一个复合语句在语法上等同于一个**独立的语句**。因此，凡是单个语句(如表达式语句)能够出现的地方都可以出现复合语句
  - 复合语句是可以**嵌套**的：复合语句作为一个语句又可以出现在**其他复合语句的内部**
  - 复合语句是以右花括号为结束标志，因此，在复合语句右括号的后面**不必加分号**，但在复合语句内的最后一个非复合语句须**以分号作为结束符**

语句和程序结构：程序设计语句

- 例5-1：复合语句的嵌套

复合语句

```
1  #include <stdio.h>
2  void main(){
3    int y;
4    y=100;
5    {
6      int x;
7      x = 20;
8      {
9        int a;
10       a = y;
11       printf("a=%d\n", a);
12       printf("x=%d\n", x);
13     }
14   }
15   printf("y=%d\n", y);
16 }
```

语句和程序结构：程序设计语句

- 例5-2：变量的生命周期
  - **函数体**也可以看成是一个**复合语句**，它是最外层的复合语句
  - 在复合语句的嵌套结构中，一个**复合语句内所进行的说明**只适合于本层中该说明语句以后的部分（包括其内层的复合语句），在该**复合语句外不起作用**

```
1  #include <stdio.h>
2  void main(){
3    int y,x;
4    y=100, x=50;
5    {
6      int x;
7      x = 20;
8      printf("x=%d\n", x);
9    }
10   printf("y=%d\n", y);
11   printf("x=%d\n", x);
12 }
```

复合语句

语句和程序结构：程序设计语句

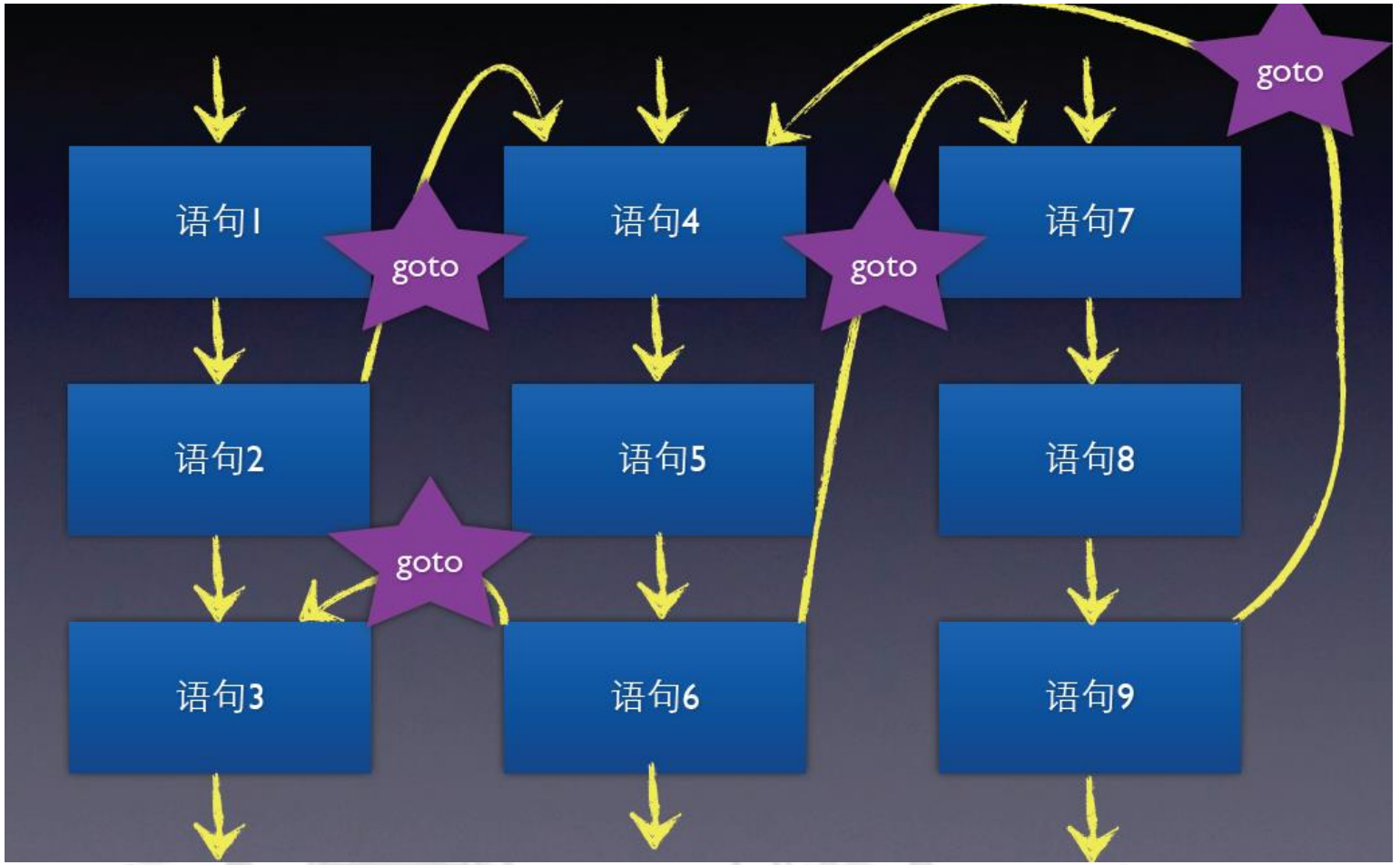
- 例5-3：变量的**掩蔽现象**
  - 在复合语句的嵌套结构中，有时在内层与外层定义了**同名的变量**
  - 这种情况按照**局部优先的原则**，内层复合语句中的变量**掩蔽外层**
  - 复合语句的同名变量，直到内层复合语句结束，**外层**复合语句的同名变量**才可以访问到**
  - **内层**复合语句中对内层定义的变量的执行结果也**不带回到外层**

```
1  #include <stdio.h>
2  void main() {
3    int x,y;
4    x=10;
5    y=100;
6    { int x;
7      x = 20;
8      printf("y=%d\n", y);
9      printf("x=%d\n", x);
10   }
11   printf("x=%d\n", x);
12 }
```

```
y = 100
x = 20
x = 10
请按任意键继续……
```

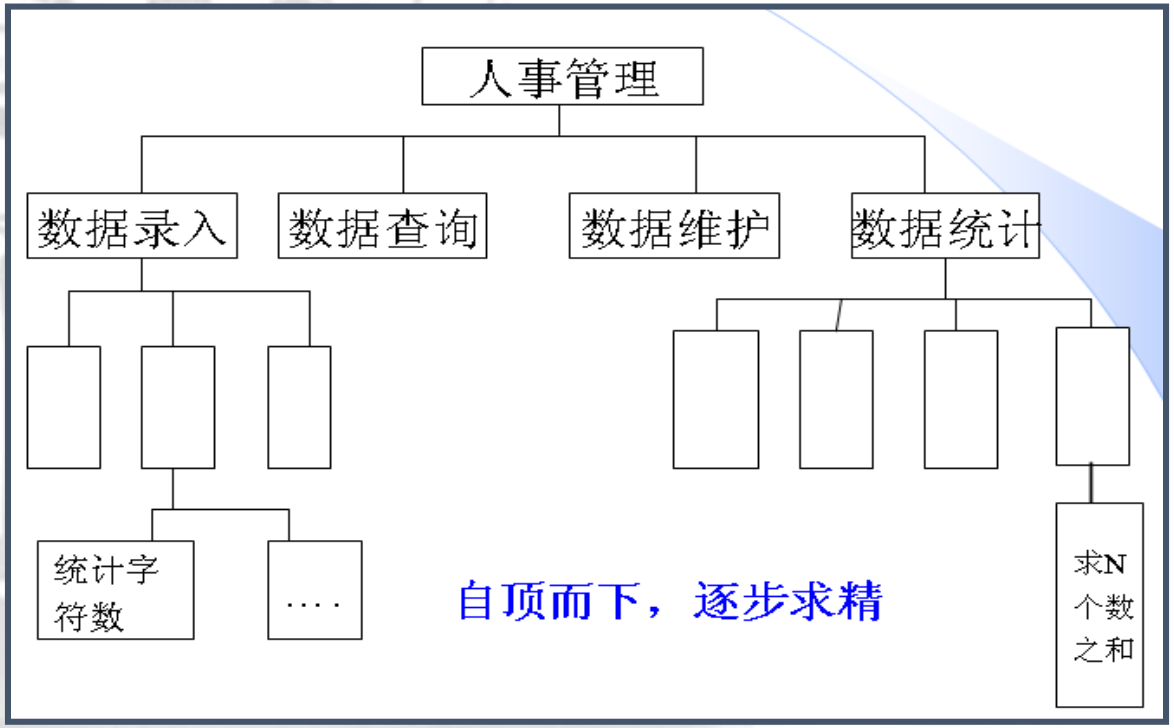
语句和程序结构：非结构化语句

- **goto语句**构成了典型的**非结构化程序**
- 非结构化程序造成程序**冗余**和**繁杂**



语句和程序结构：结构化语句

- 1968年，E. W. Dijkstra在文章"*Go to statement considered harmful* "中提出，**反对使用goto**，并提出**结构化程序设计**思想
  - 谨慎使用goto语句
  - 自顶向下设计将问题逐步分解
  - 单入口、单出口



语句和程序结构：基本语句结构

- **顺序结构**、**分支结构**和**循环结构**可以组合实现**任何程序**

- 顺序结构：无控制语句，按书写（存储）**顺序来执行**
- 分支结构：**if**语句；**switch**语句
- 循环语句：**for**语句；**while**语句；**do-while**语句

5.2 顺序结构设计

- 基本概念
- 顺序结构举例

顺序结构设计：基本概念

- 顺序结构程序指：语句执行时按照程序**存储顺序**执行的结构
  - **复合语句**：用大括号括起来的一组语句，例如

```
{     t = x;
      x = y;
      y = t; }
```

- 先执行语句1，然后顺次执行语句2和语句3

顺序结构设计：举例

- 例5-4：编写一个能将三位整数颠倒过来的程序。
  - 思路：先取出个位数，再取出十位数，最后取出百位数，然后按个位、十位、百位的顺序输出

```
1  #include <stdio.h>     //包含头文件"stdio.h"
2  void main(){           //定义main函数，这是程序的主体
3    int n=789;           //定义int型数n
4    int units, tens, hundreds;
5    units = n%10;
6    tens = (n%100)/10;
7    hundreds = n/100;
8    printf("The reverse of %3d is: %d%d%d\n", n, units, tens, hundreds);
9                         //在屏幕上打印信息
10 }
```

```
The reverse of 789 is: 987
请按任意键继续……
```

顺序结构设计：举例


- 例5-5：求一元二次 $ax^2+bx+c=0$方程的根的程序。 a,b,c的值由键盘输入，且$b^2-4ac>0$

  - 输入a,b,c的值，计算b^2-4ac，计算两个根，输出两个根

```
1  #include <stdio.h>     //包含头文件"stdio.h"
2  #include <math.h>
3  void main(){           //定义main函数，这是程序的主体
4    float a,b,c,p,q,disc,x1,x2;
5    scanf("%f%f%f",&a,&b,&c);
6    disc = b*b-4*a*c;
7    p = -b/(2*a);
8    q = sqrt(disc)/(2*a);
9    x1 = p+q;
10   x2 = p-q;
11   printf("x1=%5.2f\t x2=%5.2f\n",x1, x2);
12 }
```

```
1 5 2
x1=-0.44        x2=-4.56
请按任意键继续……
```

课堂练习：顺序结构设计

- 练习5-1：顺序执行下列语句后，a和b的值分别是什么？

```
1  #include <stdio.h>     //包含头文件"stdio.h"
2  void main() {          //定义main函数
3    int a,b;
4    scanf("%f%f",&a,&b);
5    a = a + b;
6    b = a - b;
7    a = a - b;
8  }
```

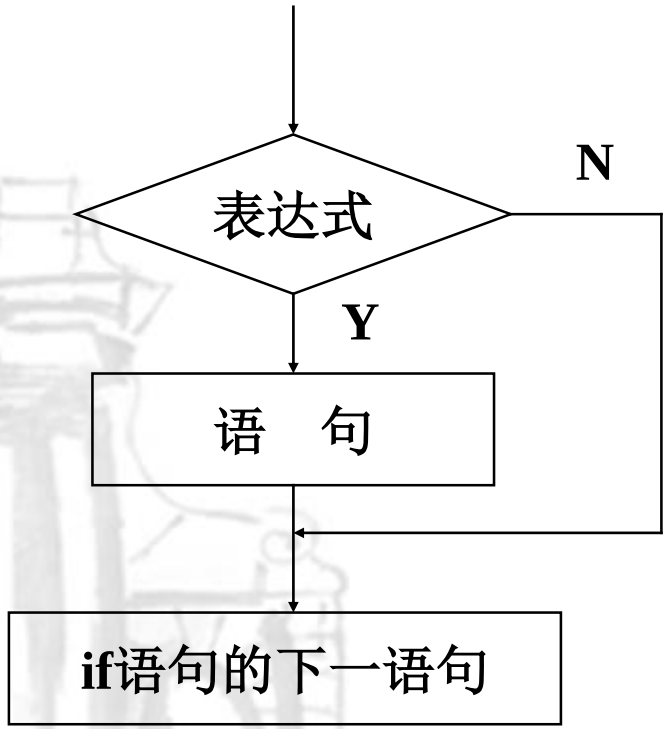
这段代码实际上在没有引入新的变量的前提下，实现了**a与b的数值交换**


5.3 分支结构设计

- if语句
  - ✓ 简单if语句
  - ✓ if-else语句
- 条件运算符
  - ✓ 介绍
  - ✓ 程序举例
- switch结构
  - ✓ 基本概念
  - ✓ 程序举例
- 分支结构应用

分支结构设计：if 语句

- if 语句的形式为: **if (表达式) 语句**
- if 语句的功能
  - 若表达式值为真(或非0)，则执行表达式后面的语句，执行完该语句后继续执行if语句后的语句
  - 若表达式值为假(0)，则不执行表达式后面的语句而直接执行if语句后的语句
  - 如果"表达式"后面的是复合语句，则要用一对花括号{ }括起来



分支结构设计：if 语句

- 例5-6：计算分段函数的值

$$y=\begin{cases}-2, & x<0\\ 2, & x\geq 0\end{cases}$$

```
1  #include <stdio.h>          //包含头文件"stdio.h"
2  void main(){                //定义main函数，这是程序的主体
3    int x,y;                  //定义int型数x,y
4    printf("Input x:\n");     //在屏幕上打印提示信息"input x"
5    scanf("%d",&x);           //输入x的值，注意其中细节
6    y = -2;                   //y=-2
7    if(x>=0) y = 2;           //若x>=0，则y=2
8    printf("y = %d\n", y);    //打印y的值
9  }
```

```
Input x:
5
y = 2
请按任意键继续……
```

分支结构设计：if 语句

- 例5-7：if+复合语句
  - 在这个程序段中，首先为变量x与y赋值3与4，然后判断变量x与y的大小
  - 若x的值小于y的值，则将变量x与y的值修改为1与2
  - 注意，if语句的作用范围**只涵盖后面一句**或者**一个复合语句**

```
1  #include <stdio.h>      //包含头文件"stdio.h"
2  void main(){            //定义main函数，这是程序的主体
3    int x,y;              //定义int型数x,y
4    x = 3;                                  //x设为3
5    y = 4;                                  //y设为4
6    if (x<y) {x = 1; y = 2;}                //选择判断
7    printf("x = %d, y = %d\n", x, y); //打印x,y的值
8  }
```

```
x = 1, y = 2
请按任意键继续……
```

分支结构设计：if 语句

- 例5-8：if约束范围辨析
  - 不要**误认为**：只有当a<b为真时，才执行x=1; y=2
  - y=2; 虽然写在if条件的后面，根本不受if条件约束，无论a<b值为真还是假，**都会被执行**
  - if条件**只约束x=1;** 当a<b为真时，才执行x=1;语句，否则x=1;语句不会被执行

```
1  #include <stdio.h>      //包含头文件
2  void main(){            //定义main函数
3     int x,y,a,b;         //定义int型
4     x = 3;               //x设为3
5     y = 4;               //y设为4
6     a = 1;               //a设为1
7     b = 2;               //b设为2
8     if(a<b) x = 1;y = 2;//选择判断
9     printf("x = %d, y = %d\n", x, y);
10 }
```

```
x = 1, y = 2
请按任意键继续……
```

分支结构设计：if 语句

- if判断条件的正确性
  - 在使用if语句时，一定要注意**逻辑表达式**的正确写法
    - 特别是在进行数值型数据比较时，一定要注意小心使用**等于比较运算符**的使用
  - 由于计算机中的实数一般都是**近似的**，对实数进行不同的运算过程其结果可能是不同的，它们之间有一定的**误差**
    - 因此，对于理论上应该相等的两个实数，在用等于运算符"=="进行比较时，结果可能是**不相等的**
  - 简单来说，两个浮点型变量：x=1.0000000000000000和y=0.9999999999999999，则**x和y并不相等**

分支结构设计：if 语句

- 例5-9：if判断条件的正确性
  - 浮点数**不能用等于号判断相等**

```
1  #include <stdio.h>     //包含头文件"stdio.h"
2  void main(){           //定义main函数
3    int flag = 0;        //定义标志变量
4    double x,y;          //定义double型数x,y
5    x = 1.0;             //x设为1.0
6    y = 0.0;             //y设为0.0
7    y = y + 0.1; y = y + 0.1; y = y + 0.1; y = y + 0.1;
8    y = y + 0.1; y = y + 0.1; y = y + 0.1; y = y + 0.1;
9    y = y + 0.1; y = y + 0.1;
10   if (x==y) flag = 1;          //x和y相等时flag为1
11   printf("flag = %d\n", flag); //打印flag的值
12 }
```

```
flag = 0
请按任意键继续……
```

分支结构设计：if 语句

- 例5-9：if判断条件的正确性
  - 用**误差范围来判断**

```
1  #include <stdio.h>     //包含头文件"stdio.h"
2  #include <math.h>      //包含头文件"math.h"
3  void main(){           //定义main函数
4    int flag = 0;        //定义标志变量
5    double x,y;          //定义double型数x,y
6    x = 1.0;             //x设为1.0
7    y = 0.0;             //y设为0.0
8    y = y + 0.1; y = y + 0.1; y = y + 0.1; y = y + 0.1;
9    y = y + 0.1; y = y + 0.1; y = y + 0.1; y = y + 0.1;
10   y = y + 0.1; y = y + 0.1;
11   if (fabs(x-y)<1.0e-10) flag = 1; //flag为1条件
12   printf("flag = %d\n", flag);     //打印flag的值
13 }
```

```
flag = 1
请按任意键继续……
```

分支结构设计：if 语句


- 例5-10，if语句**不利于实现多路**分支结构
  - 从键盘读入一个成绩，如果成绩在85～100分之间，输出"Excellent!"
  - 如果成绩在70～84分之间，输出"Good!"
  - 如果成绩在60～69分之间，输出"Pass!"
  - 如果成绩在60分以下，输出"No pass!"

```
1  #include <stdio.h>          //包含头文件"stdio.h"
2  void main(){                //定义main函数，这是程序的主体
3    float grade;              //定义float型数grade
4    printf("Input grade:"); //打印输入提示
5    scanf("%f", &grade);      //输入分数
6    if(grade>=85.0) printf("Excellent!\n");
7    if(grade>=70.0 && grade<85.0) printf("Good!\n");
8    if(grade>=60.0 && grade<70.0) printf("Pass!\n");
9    if(grade<60.0) printf("No pass!\n");
10 }
```

写了4次if语句冗余！

分支结构设计：if-else 语句

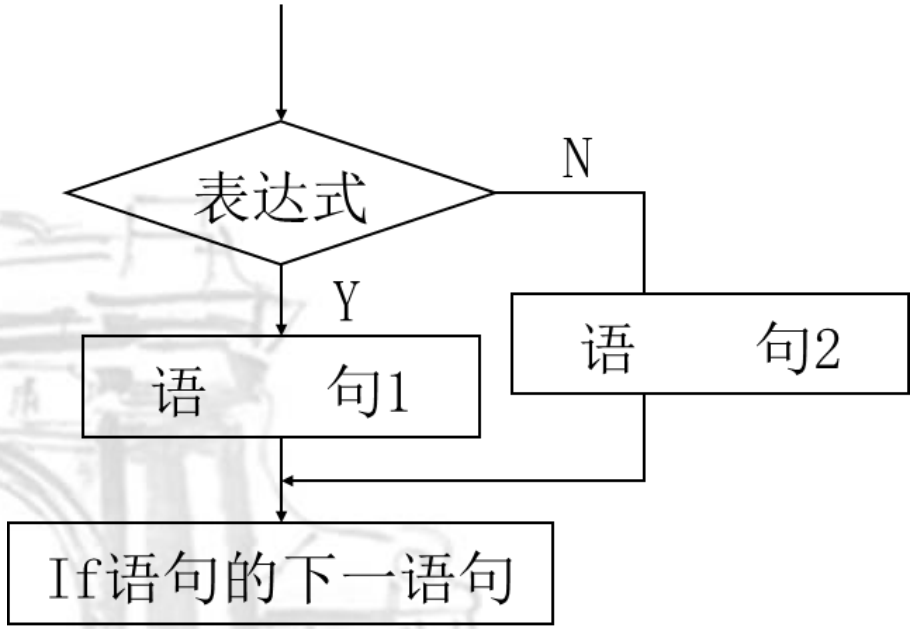
- if-else语句形式

```
if (表达式) 语句1;
else 语句2;
```

- if-else的功能
  - 若表达式值**为真(非0)**，则**执行语句1**，否则**执行语句2**。其中语句1与语句2均可以是**复合语句**
  - 使用else进行下一步的执行，但else本身**不能单独存在**
- 为了提高程序的可读性，使程序看起来逻辑更加清晰，最好将判断后的执行语句写成**复合语句**{...}

分支结构设计：if-else 语句

- if-else执行流程
  - 首先**计算表达式**的值
  - 若为**真(非0)**，则**执行语句1**，并跳转到if语句的**下一语句**
  - 若为**假(0)**，则**执行语句2**，并跳转到if语句的**下一语句**



- 一个小例子
  - if(5>3) x=5; else x=3; 则x为多少？

    结果：x=5

分支结构设计：if-else 语句

- 多路分支结构
  - if-else结构可以实现两路分支选择结构
  - C语言允许**if-else结构的嵌套**
    - 即在if-else结构中，语句1与语句2中又可以包含完整的**if语句**或**if-else结构**
    - 并且，这种**嵌套可以多层**
    - 利用if-else结构的嵌套，可以实现**多路分支选择结构**

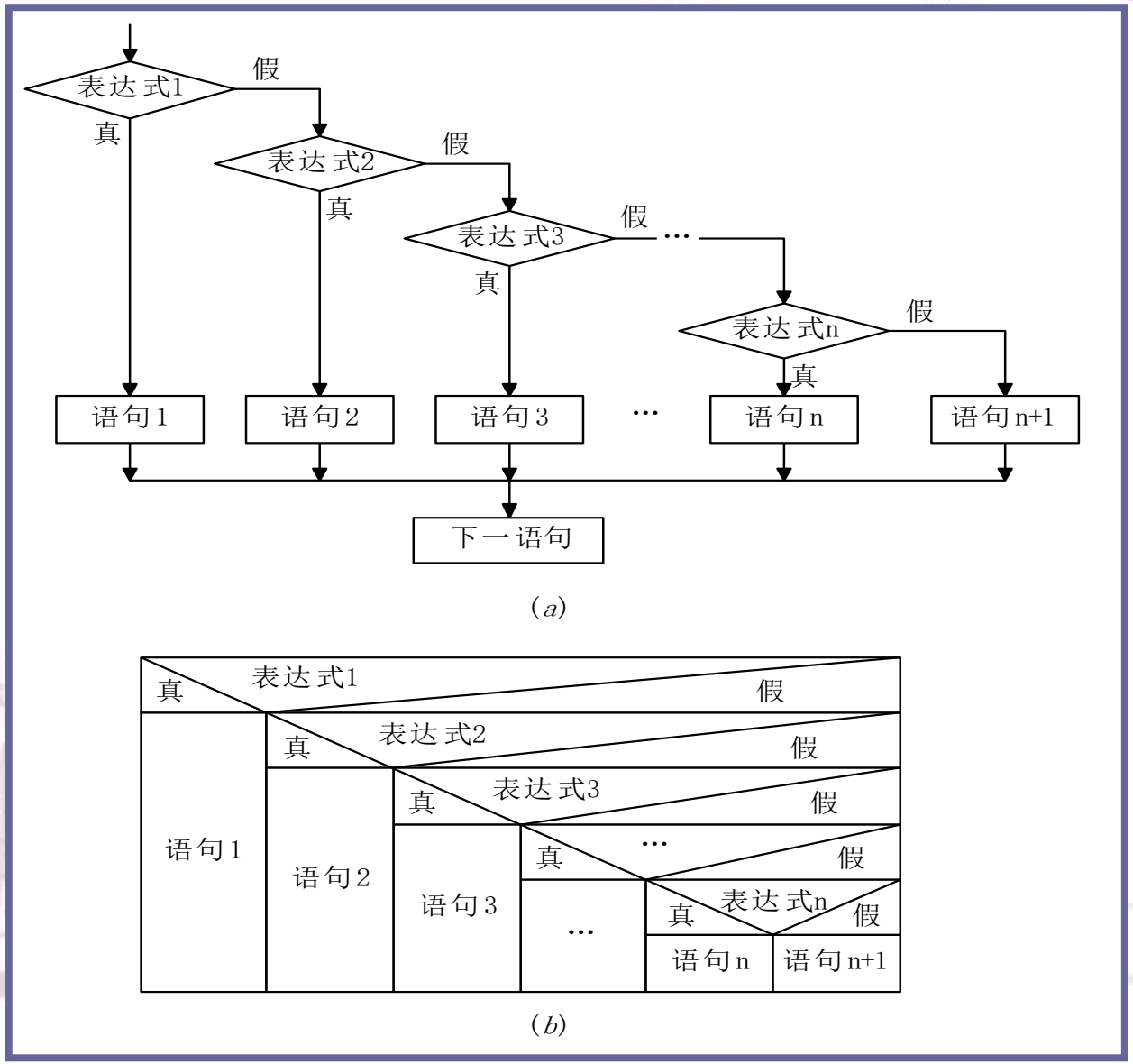
分支结构设计：if-else 语句

- if-else嵌套结构
  - 当条件1成立时，执行语句1；否则，如果条件2成立，执行语句2；否则，如果条件3成立，执行语句3 … …，**直到当条件n成立时，执行语句n，否则执行语句n+1**
  - if-else嵌套结构是编程中最常用的语法结构之一，其中**条件的设置**及**顺序**很重要

```
1  if(条件1) 语句1;
2  else if(条件2) 语句2;
3       else if(条件3) 语句3;
4            ……
5            else if(条件n) 语句n;
6                 else 语句n+1;
```

分支结构设计：if-else 语句

- if-else嵌套结构图
  - 分别用两种结构表示



分支结构设计：if-else 语句

- 例5-11：多路分支结构
  - 采用**if-else 嵌套结构**重写例5-10

```
1  #include <stdio.h>     //包含头文件"stdio.h"
2  void main(){           //定义main函数，这是程序的主体
3    float grade;         //定义float型数grade
4    printf("Input grade:"); //打印输入提示
5    scanf("%f", &grade);    //输入分数
6    if(grade>=85.0) printf("Excellent!\n");
7    else if(grade>=70.0) printf("Good!\n");
8    else if(grade>=60.0) printf("Pass!\n");
9    else if(grade<60.0)  printf("No pass!\n");
10 }
```

```
Input grade: 90
Excellent!
请按任意键继续……
```

分支结构设计：if-else 语句

- 例5-12，**条件设置顺序影响执行效率**
  - 计算并输出下列分段函数值，其中x从键盘输入

$$y=\begin{cases}0, & x<-10\\ 2x+20, & -10\le x<0\\ 20, & 0\le x<20\\ 30-0.5x, & 20\le x<40\\ 50-x, & 40\le x<50\\ 0, & 50\le x\end{cases}$$

  - 观察到：如果输入的x值大于等于50,只需要判断1次；但如果输入的x值小于-10,则需要判断5次

分支结构设计：if-else 语句

- 条件设置顺序影响执行效率：对比两种实现程序

实现方法A

```
1  #include <stdio.h>
2  void main(){
3    float x,y;
4    printf("Input x:");
5    scanf("%f", &x);
6    if (x>=50.0) y = 0.0;
7    else if (x>=40.0)  y = 50-x;
8    else if (x>=20.0)  y = 30-0.5*x;
9    else if (x>=0.0)   y = 20.0;
10   else if (x>=-10.0) y = 2*x+20;
11   else y = 0.0;
12   printf("x = %f, y = %f\n", x, y);
13 }
```

实现方法B

```
1  #include <stdio.h>
2  void main() {
3    float x,y;
4    printf("Input x:");
5    scanf("%f", &x);
6    if (x>=20.0){
7      if (x>=50.0)          y = 0.0;
8      else if (x>=40.0)     y = 50-x;
9      else                  y = 30-0.5*x;
10   }else{
11     if (x>=0.0)           y = 20.0;
12     else if (x>=-10.0)    y = 2*x+20;
13     else                  y = 0.0;
14   }
15   printf("x = %f, y = %f\n", x, y);
16 }
```

分支结构设计：if-else 语句

- 条件设置顺序影响执行效率
  - 对于实现**方法A**，如果输入的x值大于等于50,只需要判断1次；但如果输入的x值小于-10,则**需要判断5次**
  - 对于实现**方法B**，无论输入的x值是什么，**最多需要判断3次**即可得出结果
  - 这提示我们，多运用程序设计的技巧，可以提高程序的**执行效率**
  - 建议：在处理多路分支选择时，应尽量将出现**几率高的条件写在前面**，以**二分法设置判断顺序**，提高执行效率
  - 注意：将if和else后面的执行内容，按照{…}写成复合语句，从而提高程序的**可读性**

课堂练习

- 练习5-2：应用表达式实现对闰年的判断
  - 题目：编程输入公元年号，并判断是否是闰年
  - 思路：闰年的2个条件：**能被4整除**，**但不能被100整除**；**能被100整除**，**又能被400整除**
  - 程序设计
    - 读入年份，首先判断是否能被4整除
    - 如果可以被4整除，判断是否可以被100整除。若否，则为闰年；若是，继续判断能否被400整除，若是则为闰年，否则为非闰年
    - 如果不可以被4整除，非闰年

课堂练习

- 练习5-2

```
#include <stdio.h>
void main(){
  int leap, year;
  printf("Input year:");
  scanf("%d", &year);
  if (year%4==0){
    if (year%100==0){
      if (year%400==0) leap = 1;
      else leap = 0;
    }
    else leap = 1;
  }
  else
    leap = 0;
  if (leap) printf("%d is", year);
  else printf("%d is not", year);
  printf( " a leap year.\n");
}
```

```
Input year: 2000
2000 is a leap year.
请按任意键继续……
```

还可以用上一讲中介绍的逻辑表达式((year%4==0 && year%100!=0) || year%400==0)，然后使用if语句对闰年进行判断

分支结构设计：if-else 语句

- 总结
  - 在分类情况比较多的时候采用if-else结构
  - if-else结构中的语句1与语句2都可以是复合语句{…}
  - if-else结构中，语句1与语句2都可以是空语句（;）
  - 如果在else前面有多个if语句，则else与同层最近的if配对，与书写方式无关
  - 合理设置判断顺序可以提高程序执行效率

分支结构设计：条件运算符

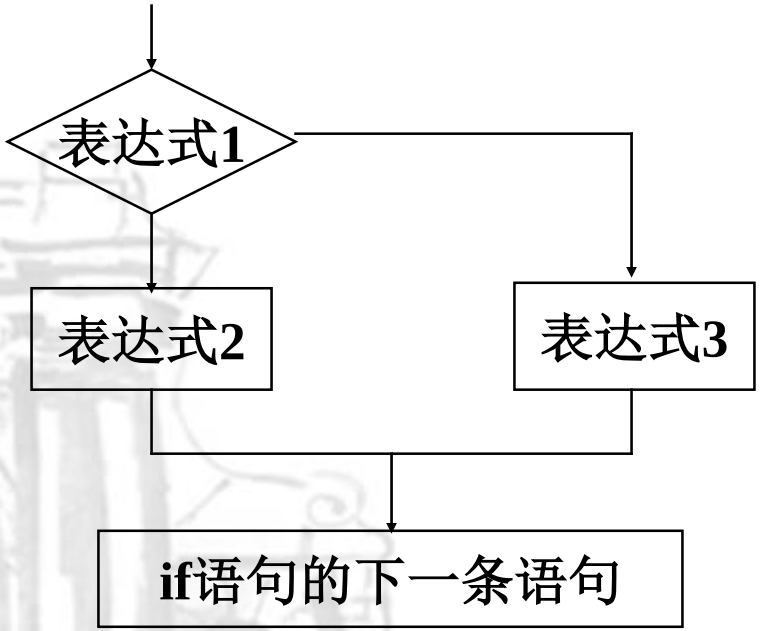
- 条件运算符的表达式

  表达式1？表达式2：表达式3

- 条件运算符的功能
  - 方便、紧凑地实现一个**简单的分支语句**
- 特点
  - 条件运算符优先级要比**赋值运算符**要**高**
  - 条件运算符的优先级比**关系运算符**、**算术运算符**都要**低**
  - 条件运算符的结合方向为"**从右到左**"

分支结构设计：条件运算符

- 条件运算符执行流程
  - 首先计算表达式1的值
  - 若为真（非0），则计算表达式2，其整个表达式的值是表达式2的值
  - 若表达式1的值为假(0)，则计算表达式3，其整个表达式的值是表达式3的值



- 一个小例子
  - 若x=0?4:0?3:5，则x为多少？

    x=（0?4:（0?3:5））=0?4:5=5

分支结构设计：条件运算符

- 例5-13：条件运算符精简代码
  - 从键盘输入一个x，计算并输出下列分段函数值：

$$y=\begin{cases}x^2-1, & x<0\\ x^2+1, & x\geq 0\end{cases}$$

if语句需要两句 → **条件语句仅需一句**

```c
#include <stdio.h>
void main(){
  float x,y;
  printf("Input x:");
  scanf("%f", &x);
  if(x<0) y = x*x-1;
  else y = x*x+1;
  printf("y = %f\n", y);
}
```

```c
#include <stdio.h>
void main(){
  float x,y;
  printf("Input x:");
  scanf("%f", &x);
  y=(x<0)?(x*x-1):(x*x+1);
  printf("y = %f\n", y);
}
```

课堂练习

- 练习5-3：条件运算符和字符运算
  - 从键盘输入一个字符，如果输入的是英文**大写字母**，则将它**转换成小写字母**后输出，否则输出原来输入的字符

```
1  #include <stdio.h>
2  void main(){
3    char ch;
4    printf("Input ch:");
5    scanf("%c", &ch);
6    ch = (ch>='A' && ch<='Z') ? ch -'A'+'a': ch;
7    printf("%c\n", ch);
8  }
```

```
Input ch: D
d
请按任意键继续……
```

分支结构设计：条件运算符

- 例5-14：编程找出3个数中最大者
  - 将前两个数中最大的数放入**临时变量**temp中
  - 再将temp和最后一个数进行比较，较大的数即为max变量

```
1  #include <stdio.h>
2  void main(){
3     int a,b,c,temp,max;
4     printf("Input a,b,c:");
5     scanf("%d %d %d", &a, &b, &c);
6     temp = (a>b) ? a : b;
7     max = (temp>c) ? temp : c;
8     printf("max=%d\n", max);
9  }
```

```
Input a,b,c: 3 10 8
max=10
请按任意键继续……
```

分支结构设计：switch 结构

- switch语句表达式
  - case命令用来描述**判断条件**
  - 语句n用来执行**判断后的命令**
  - default用来做**备选条件**

```
1  switch(表达式){
2     case 常量表达式1: 语句1;
3     case 常量表达式2: 语句2;
4     …
5     case 常量表达式n: 语句n;
6     default: 语句n+1;
7  }
```

- switch语句功能
  - 对于大量判断条件，以及对应条件下不同的执行语句，采用switch语句可以结构清晰、方便地**实现分支功能**

分支结构设计：switch 结构

- switch语句特点
  - switch结构中的表达式、常量表达式1、…、常量表达式n**必须是整型**或**字符型**
  - 同一个switch结构中的常量表达式的值**必须互不相同**，否则就会**出现编译错误**，因为"表达式"的同一个值对应多种执行方案，这是错误的
  - 在switch结构中，case 与default 的**顺序可以任意**，各case之间的**顺序也可以任意**

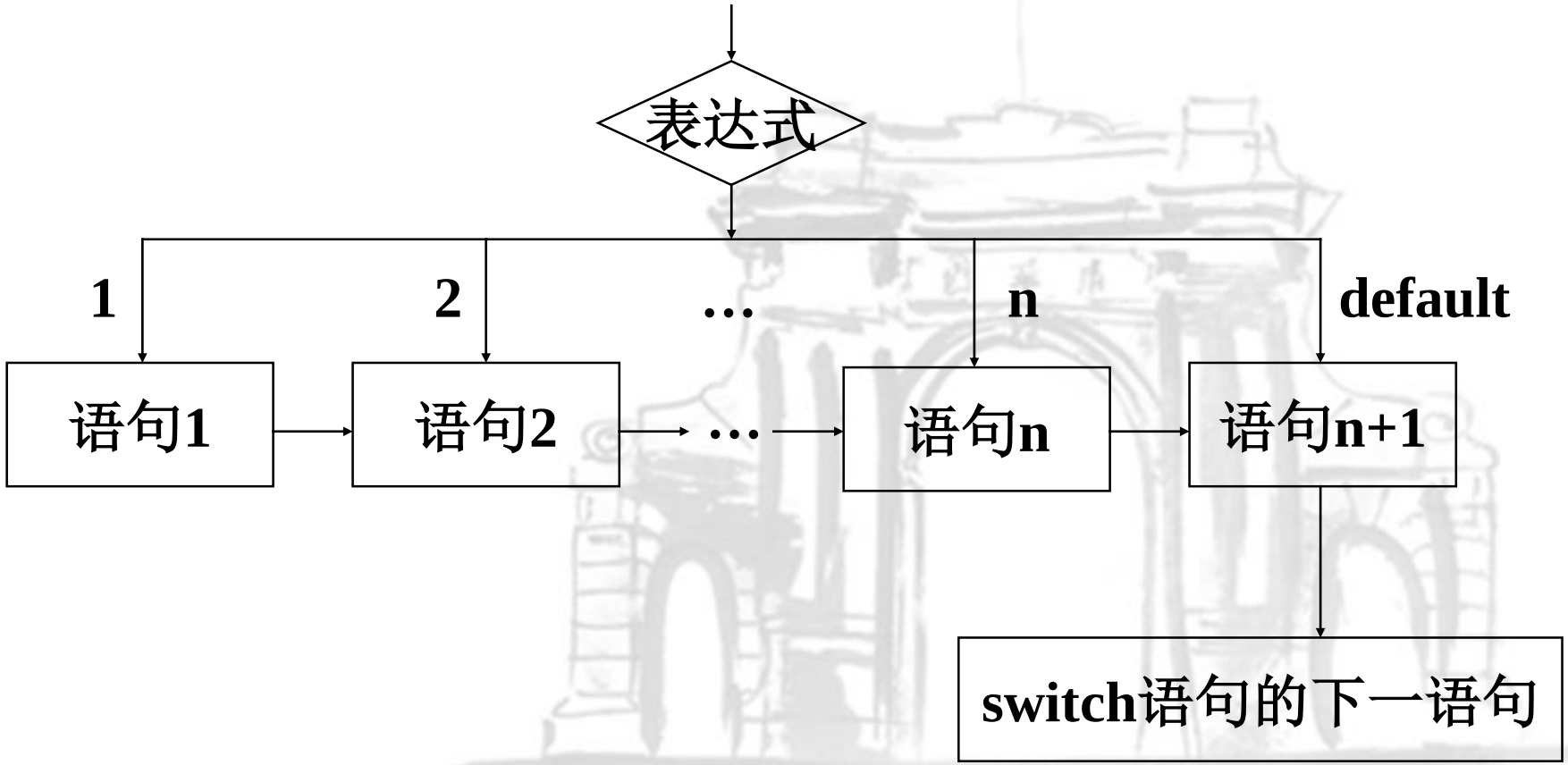
分支结构设计：switch 结构

- switch语句特点
  - 当执行完某case的语句后，如果没有遇到终止语句，将**顺序继续**执行后面case或default的语句，**直到遇到break语句**才退出整个switch结构
  - 如果**没有default**且"表达式"值**不等于**任何case 后常量表达式的值，**则不做任何事情**，直接退出switch结构

switch语句就像坐电梯，每个case语句后面的标号相当于一个按钮，你要在哪一层，就按那一层按钮；如果要下去就加break

分支结构设计：switch 结构

- switch语句结构



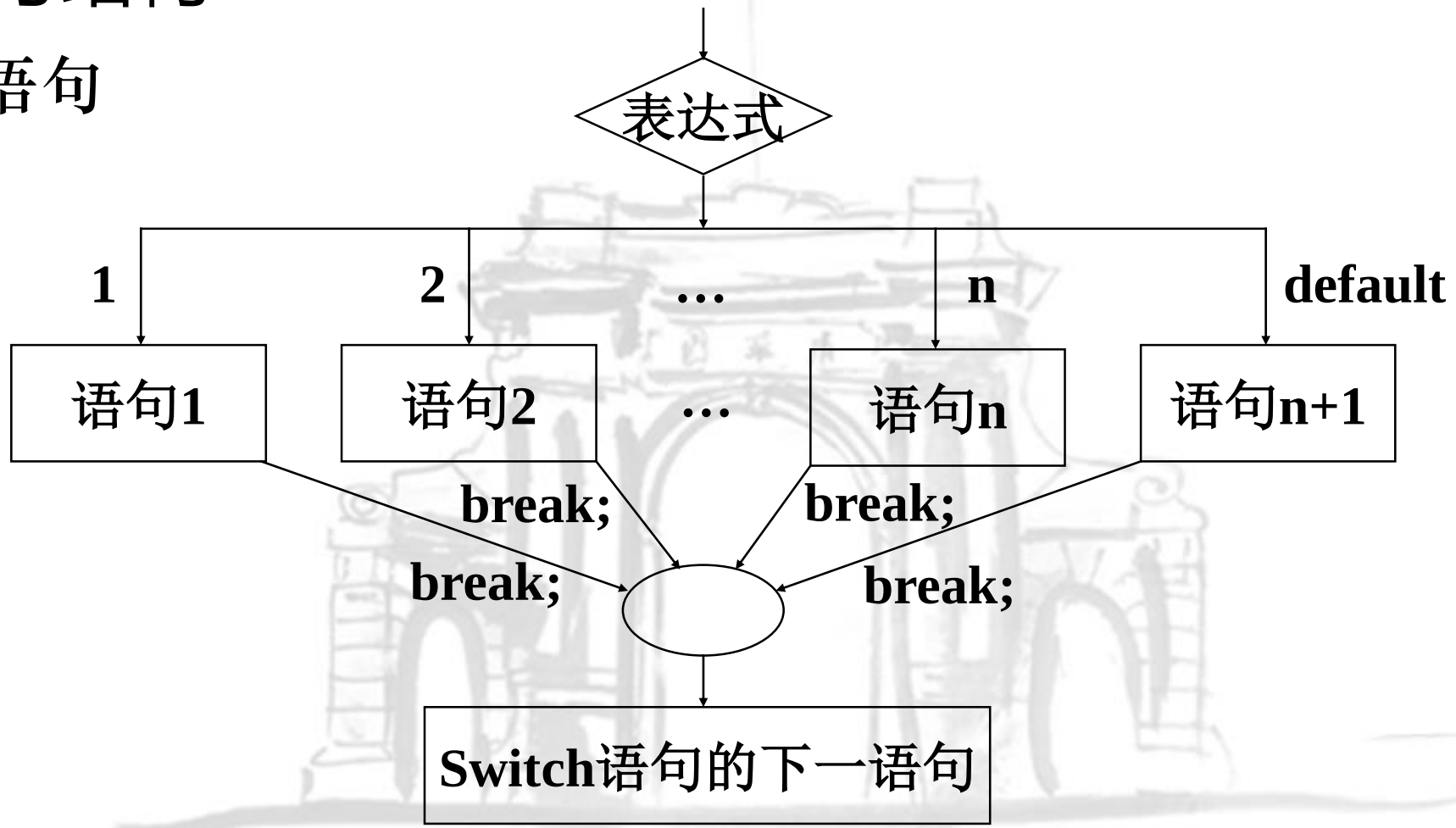
分支结构设计：switch 结构

- break语句表达式
  - 形式为 break;
- break语句功能
  - 作用是从switch、for、while或do-while语句中**跳转出来**，终止这些语句的执行，把控制转到被中断的循环语句或switch语句之后去执行
  - 单独使用break语句是没有意义的，一般地，它都**与循环语句或switch语句连用**

```
1  switch(表达式){
2     case 常量表达式1: 语句1; break;
3     case 常量表达式2: 语句2; break;
4     …
5     case 常量表达式n: 语句n; break;
6     default: 语句n+1;
7  }
```

分支结构设计：switch 结构

- switch语句结构
  - 含break语句



分支结构设计：switch 结构

- 例5-15：break作用辨析
  - 从键盘读入一个成绩，如果成绩大于90，则输出"A"
  - 如果成绩在80～90分之间，则输出"B"
  - 如果成绩在70～80分之间，则输出C
  - 如果成绩在60~70之间，则输出"D"
  - 如果成绩小于60，则输出"E"

```
1  #include <stdio.h>
2  void main(){
3    float grade;
4    printf("Input grade:");
5    scanf("%f", &grade);
6    switch(int(grade/10)){
7      case 10:
8      case 9: printf("A!\n"); break;
9      case 8: printf("B!\n"); break;
10     case 7: printf("C!\n"); break;
11     case 6: printf("D!\n"); break;
12     default: printf("E\n");
13   }
14 }
```

分支结构设计：switch 结构

- 例5-15：break作用辨析
  - 注意"break;"语句**非常重要**，如果将其去掉，则：

```
1  #include <stdio.h>
2  void main(){
3    float grade;
4    printf("Input grade:");
5    scanf("%f", &grade);
6    switch (int(grade/10)){
7      case 10:
8      case 9: printf("A!\n");
9      case 8: printf("B!\n");
10     case 7: printf("C!\n");
11     case 6: printf("D!\n");
12     default: printf("E!\n");
13   }
14 }
```

```
Input grade: 100
A!
B!
C!
D!
E!
请按任意键继续……
```

分支结构设计：switch 结构

- 例5-16：**多条件**判断
  - 铁路规定的购票标准为：身高1米以下的儿童免票，超过1米不足1.4米的儿童购买半票，超过1.4米的儿童购买全票。编写一个购票程序
  - 思路：设三个变量分别为：儿童的身高h，全程票价price，购票款为paying
  - 实现：可以用两种方法编写程序，其一是采用if-else结构，对不同的情况进行判断；其二是采用switch结构进行判断

分支结构设计：switch 结构

- 例5-16：多条件判断

```c
#include <stdio.h>
void main(){
  float h, price, paying;
  printf("Input high and price:");
  scanf("%f %f", &h, &price);
  if (h <= 1.0)
    paying = 0;
  else if (h <= 1.4)
    paying = price/2;
  else
    paying = price;
  printf("paying=%f\n",paying);
}
```

```c
#include <stdio.h>
void main(){
  float h, price, paying;
  int temp;
  printf("Input high and price:");
  scanf("%f %f", &h, &price);
  if (h >= 1.4)  temp = 2;
  else  temp = (int)h/1;
  switch(temp) {
    case 0: paying = 0; break;
    case 1: paying = price/2; break;
    case 2: paying = price; break;
  }
  printf("paying=%f\n",paying);
}
```

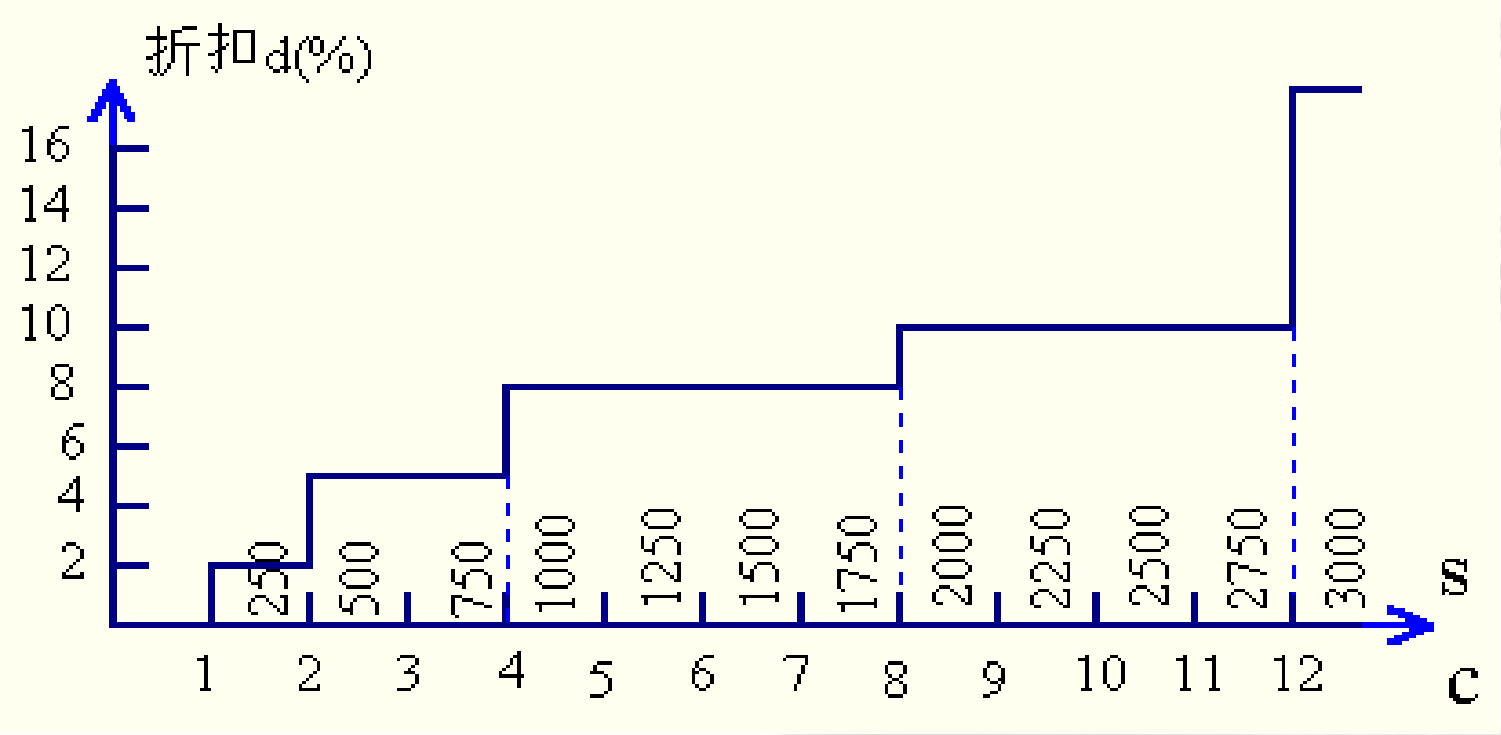
switch语句表示更加清晰！

分支结构设计：switch 结构

- 例5-17：复杂条件问题
  - 运输公司对客户计算运费，根据距离（s）给出一定折扣。折扣标准如下
    - s<250km，没有折扣；　　250≤s <500，2%折扣；
    - 500≤s<1000，5%折扣；　　1000≤s<2000，8%折扣；
    - 2000≤s<3000，10%折扣；　3000≤s，15%折扣
  - 设每公里每吨货物的基本运价为p，货物重w吨，距离s，折扣率d，则总运费f为：f = p*w*s*(1-d)
  - 折扣率的变化点均为250的倍数，用新变量c表示距离的范围
    - s<250，c = 0；　　250≤s<500，c = 1
    - 500≤s<750；c = 2；.....　3000≤s，c = 12

分支结构设计：switch 结构

- 例5-17：复杂条件问题
  - 将题目中的复杂关系用图的方式来表达



如果：S>3000,则c=12;

否则： c=s/250;

分支结构设计：switch 结构

- 例5-17：复杂条件问题
  - 方便地对十个以上条件进行罗列

```
#include <stdio.h>
void main(){
  float p, w, d, f;
  int c, s;
  scanf("%f %f %d", &p, &w, &s);
  if (s>=3000) c = 12;
  else c = (int)s/250;
  switch(c){
    case 0: d = 0; break;
    case 1: d = 2; break;
    case 2:
    case 3: d = 5; break;
    case 4:
    case 5:
    case 6:
    case 7: d = 8; break;
    case 8:
    case 9:
    case 10:
    case 11: d = 10; break;
    case 12: d = 15; break;
  }
  f = p*w*s*(1-d/100.0);
  printf("freight=%f\n",f);
}
```

分支结构设计：分支结构应用

- 例5-18：解二元一次方程组
  - 设一元二次方程为$Ax^2+Bx+C=0$，请求解该方程
- 思路：求解一元二次方程要考虑各种特殊情况，包括
  - 首先考虑**系数A是否等于0**：如果A=0，此时方程变为$Bx+C=0$，则还要考虑以下两种情况
    - 若B=0，则方程无意义(因为A与B都为0)，输出"ERR"，结束
    - 若B≠0，则方程只有一个实根，即输出x=-C/B，结束
  - 如果**A≠0**，需要考虑以下**两种情况**

分支结构设计：分支结构应用

- 若B=0，此时方程变为$Ax^2+C=0$ 在这种情况下，如果A与C异号，则方程有两个实根为$x_{1,2}=\pm\sqrt{-C/A}$；如果A与C同号，则方程有两个虚根为$x_{1,2}=\pm j\sqrt{C/A}$，其中$j=\sqrt{-1}$
- 若$B\neq 0$，此时方程变为$Ax^2+Bx+C=0$ 在这种情况下，如果C=0，则方程变为$Ax^2+Bx=0$ 两个实根分别为$x_1=0$和$x_2=-B/A$；如果$C\neq 0$，计算判别式$D=B^2-4AC$，再考虑以下两种情况

如果$D\geq 0$，则表示方程有两个实根，公式为：

$$x_{1,2}=(-B\pm\sqrt{-D})/(2*A)$$

如果$D<0$，则表示方程有两个共轭复根，公式为：

$$x_{1,2}=(-B\pm j\sqrt{-D})/(2*A)$$

分支结构设计：分支结构应用

- **NS流程图**可以用来清晰地展示分支结构

<table>
<tr><td colspan="6">输入系数A, B, C</td></tr>
<tr><td colspan="2">Yes</td><td colspan="4">A=0 No</td></tr>
<tr><td>Yes</td><td>B=0 No</td><td colspan="4">Yes B=0 No</td></tr>
<tr><td rowspan="3">输出"ERR"</td><td rowspan="3">x1=-C/B
输出x1</td><td colspan="2">D=-C/A</td><td>Yes</td><td>C=0 No</td></tr>
<tr><td>Yes</td><td>D>=0 No</td><td rowspan="2">x1=0
x2=-B/A</td><td>D=B*B-4*A*C</td></tr>
<tr><td>x1=sqrt(D)
x2 = -x1</td><td>x1=-j*sqrt(-D)
x2 = -x1</td><td>Yes D>=0 No
Yes B>0 No
x1=-(B + sqrt(D))/(2*A)
x2 = C/(x1*A)
x1=-(B - sqrt(D))/(2*A)
x2 = C/(x1*A)
P=-B/(2*A)
Q=sqrt(-D)
x1=P + jQ
x2=P - jQ</td></tr>
<tr><td colspan="6">输出x1, x2</td></tr>
</table>

分支结构设计：分支结构应用

- 求解一元二次方程

```c
#include <stdio.h>              //包含头文件"stdio.h"
#include <math.h>               //包含头文件"math.h"
void main() {
  double a,b,c,d,x1,x2,p;
  printf("input a, b, c:");
  scanf("%lf %lf %lf", &a, &b, &c);
  if (fabs(a)<=1e-6) {  //一次方程，1e-6仅为示例
     if (fabs(b)<=1e-6) printf("ERR\n");
    else printf("x=%f\n",-c/b);
  }
  else if (fabs(b)<=1e-6) { //Ax^2+C=0
    d = c/a;
    if (d<=0.0)
      printf("x1=%f,x2=%f\n",sqrt(-d),-sqrt(-d));
    else
      printf("x1=+j%f,x2=-j%f\n",sqrt(d),sqrt(d));
  }
  else if (fabs(c)<=1e-6)          //Ax^2+Bx=0
    printf("x1=0.0,x2=%f\n",-b/a);
  else {                           //Ax^2+Bx+C=0
    d = b*b-4*a*c;
    if (d>=0.0) {
      d = sqrt(d);
      if (b>0.0) x1 = (-b-d)/(2*a);
      else x1 = (-b+d)/(2*a);
      x2 = c/(a*x1);
      printf("x1=%f,x2=%f\n",x1,x2);
    }
    else {
      d = sqrt(-d)/(2*a);
      p = -b/(2*a);
      printf("x1=%f+j%f,x2=%f-j%f\n",p,d,p,d);
    }
  }
}
```

分支结构设计：分支结构应用

- 例5-19：运算符处理程序，编制一个C程序，功能是
  - 首先从键盘依次输入两个实数作为运算对象
  - 从键盘再输入一个运算符，最后输出运算结果
  - 其中运算符的符号分别为：加法运算符"+"、减法运算符"-"、乘法运算符"*"或点"."；除法运算符"/"
  - 在作除法运算时，如果第二个实数为0时，要求输出信息"error!"
  - 如果输入的运算符不是上述所定义的运算符，要求输出信息"Incorrect symbol!"

分支结构设计：分支结构应用

- 思路
  - 首先定义和读取数值变量，用于后续运算
  - 读取输入的运算符
  - 对运算符进行**分类判断**，并按照不同的情况对数值变量**进行计算**，并最终**输出结果**
- 注意
  - 在C语言中，使用scanf语句读取数据后，会有一个**回车符号残留在输入流**中，这将影响下一次字符型变量的读取。因此应该先**处理这一无用的字符**

分支结构设计

- 运算符处理程序
  - 接收字符用**getchar**
  - 先读掉回车符号**残留**
  - 判断**被除数是否为0**

```
1  #include <stdio.h>          //包含头文件"stdio.h"
2  #include <math.h>           //包含头文件"math.h"
3  void main(){
4    double x,y;  char ch;
5    printf("Input x, y:"); scanf("%lf %lf", &x, &y);
6    ch = getchar();  //读掉上次输入中最后一个回车符号
7    printf("Input ch");
8    ch = getchar();  //读输入运算符字符
9    switch(ch) {
10     case'+': printf("%f+%f=%f\n",x,y,x+y); break;
11     case'-': printf("%f-%f=%f\n",x,y,x-y); break;
12     case'*': printf("%f*%f=%f\n",x,y,x*y); break;
13     case'/': if (fabs(y)<=1e-7) printf("error!\n");
14              else  printf("%f/%f=%f\n",x,y,x/y);
15              break; //判断分母为0
16     default: printf("Incorrect symbol!\n");
17   }
18 }
```

本课总结

- **语句和程序结构**
  - 语句：语句介绍；结构化和非结构化程序
  - 基本程序结构：顺序结构、分支结构、循环结构可构成所有程序
- **顺序结构设计**
  - 顺序结构介绍和举例
- **分支结构设计**
  - if语句：简单if语句和if-else语句
  - 条件运算符：介绍和程序举例
  - switch结构：介绍和程序举例

本课作业

- 第五讲作业
  - 教材p.121~123习题2，3，7，12，18，19
  - 完成后将**word文档**或**拍照**提交到网络学堂
- 附加作业
  - 1、程设课程的评分标准是：30分平时作业，10分期中机考，20分期末机考和40分期末笔试。编写程序，输入这四部分的成绩a,b,c,d，首先判断各部分的成绩是否合法（即是否超过满分），再判断该同学是否通过了这门课程

本课作业

- 附加作业
  - 2、输入整型数X，Y，Z，判断由X，Y，Z组成的日期“X年Y月Z日”是否是合法的日期（按现行公历计算，不考虑历史上历法切换造成的一系列问题）
  - 3、循环移位加密：输入一个拉丁字母，和一个整型数n，输出该字母循环平移了n位后的结果，保持大小写不变。如：输入A，3，则输出为D；输入a，-1，输出z


THANKS