C语言

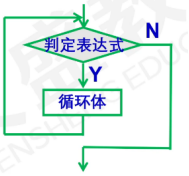
第一章：C语言基础知识

C语言基础知识 一

1. **程序**：一系列指令序列，即人机对话的语言
2. **语言分类**
    - **低级语言**：机器语言、汇编语言
    - **高级语言**：C,C++等

程序设计的三大基本结构

1. 顺序结构
    - 从上到下从左到右 

2. 选择结构
3. 循环结构
    - 

概念

1. 源程序：高级语言编写的程序
2. 目标程序：二进制代码表示的程序
3. 开发一个C程序的基本过程：
    - 编辑(.c 源程序)->编译(.obj 目标程序)->连接(.exe 可执行程序)->运行
4. C代码编译成可执行程序经过4步：
    - 预处理->编译>汇编->链接

C语言的基本结构

1. ```c
   #include <stdio.h>
   int main(){
       printf("Hello World\n");
       return 0;
   }
   ```

2. 介绍
    - 1、编译预处理命令
        - 以#开头 后面无分号
        - #include（包含）
        - #define （宏定义）
    - 2、stdio.h文件中 包含了有关
   ```

        - (scanf)、输入
        - (printf) 、输出功能的函数。
    - 3、**函数是程序的基本单位。一个程序有且只有一个main()函数**，位置任意程序执行从main开始，到main结束。
    - 4、**分号为语句的分隔符**
    - 5、大括号标识一个语句组，成对使用
    - 6、若主函数为 void 可省略 主函数还可以是int
3.
```c
#include <stdio.h>
int main(){
    int a,b,c;
    a=4;
    b=10;
    c=a+b;
    printf("%d\n",c);
    return 0;
}
```

标识符

1. 标识符：在C语言中，有许多符号的命名如变量名、函数名、数组名等，都必须遵守一定**的规则**，按此规则命名的符号**称为标识符。**
2. 标迟符的命名规则：
    - **字母、数字、下划线_构成的有限序列**
    - **以字母或者下划线开头**
    - 标识符不能包含除以外的任何特殊字符，如：%、#、逗号、空格等。
    - 标识符**必须以字母或（下划线）开头。**
    - 标识符不能包含空白字符（换行符、空格和制表符称为空白字符）
    - C语言中的某些词（例如int和float等）称为**保留字**，具有特殊意义，不**能用作标识符名。**
    - C语言**区分大小写**，因此标识符price.与标识符PRICE是两个不同的标识符。
3. 分类
    1. **关键字**(32个)：C语言规定了一批标识符，他们在C语言中代表着固定的含义，不能另做它用。

        ```
        auto break case char const continue default do double
        else enum extern float or goto if int long register return
        short signed sizeof static struct switch typedef union
        unsigned void volatile while
        ```

    2. **预定义标识符**：C语言**语法允许**用户把这类标识符另**做它用**，但是这些标识符将失去系统规定的原意。

        比如：printf define main scanf

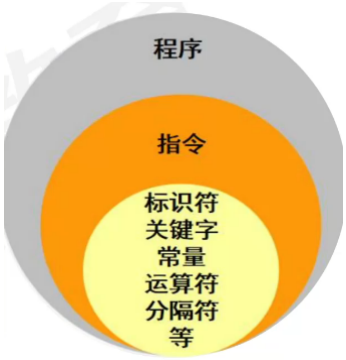
    3. **用户标识符**：由用户根据需要定义的标识符称为用户标识符

注释

- ```
  //行注释
  /*注释内容不是程序代码，是给其他人员增强理解能力加上的文字说明当程序执行时注释内容视为空 */
  ```
  块注释
  ```

二、量

一、常量

1. 常量是在程序中保持不变的量
2. 常量用于定义具有如下特点的数据
   - 在程序运行中**保持不变**
   - 在程序内部频繁使用
   - 用define关键字定义**符号常量**
   - 分类：整型常量、实型常量、字符常量、字符串常量 符号常量
     举例：1 2.5 'a' “abc”
   - 

整型常量

- 整数的表示
  1. 十进制表示：用一串连续的数字(0~9)表示十进制数。
     例如：345 3684 0 -23456 ***只有十进制负数前面加-。**
  2. 八进制表示：以数字**0**开头的一个连续数字序列，序列中只能有**0-7**这八个数字。（无负数）
     例如：045 067451等。 而：019 423 -078都是非法的八进制数。
  3. 十六进制表示：以0X或0x开头的连续数字和字母序列，序列中只能有0-9、A-F和a-f这些数字和字母，字母a、b、c、d、e、f分别对应十进制数字10、11、12、13、14、15,大小写均可。(0x里面x小写就用小写a-f 大写X就是A-F)（无负数）

实型常量

1. 小数形式
   - 由数字和小数点组成，**必须有小数点。**(整数部分，小数部分为0可省略小数点不能省略)
     例如：3.14 0.15 .56 78. 0.0
2. 指数形式
   - 以**幂**的形式表示，以字母e或E后跟一个以10为底的幂数。
   - 数字 字母+(e/E) 指数(正负)

     ***注意*：例：2.3e5、500e-2、.5E3、4.5e0，而e4、.5e3.6、.e5、e都不合法**
   - 字母e或E之前必须要有数字，且字母e或E后面的指数必须为整数，
     字母e或E的前后及数字之间不得有空格（实型变量也遵守这个
     规则)。技巧记忆：**e前e后必有数，e后必须是整数(负数0整数)！**e前为数
     例：2.3E5 500e-2 .5e3 4.5E0，而e4 .5E3.6 .E5 e都不合法。

字符常量

1. 定义：在程序中用**一对单引号把一个字符**括起来，作为字符常量。例'A','a','t','!','*','\n'等。Ascll A值65 a值97
2. 说明
   - (1)字符常量只能**用单引号**括起来，**不能用双引号**；
     如：“A”不是字符常量
   - (2)字符常量**只能包含一个字符**；如：'abc',错误。
   - (3)**区分大小写**；
     如：'A'和'a
   - (4)**字符**是**按**其代码 **ASCLL码(整数值)形式存储**，所有字符数据都作为整数来处理。因此，字符量可以**参与整数运算。**

值	符号	值	符号	值	符号
0	空字符	44	,	91	[
32	空格	45	—	92	\
33	!	46	.	93	]
34	”	47	/	94	^
35	#	48-57	0-9	95	_
36	￥	58	:	96	`
37	%	59	;	97-122	a-z
38	&	60	<	123	{
39	‘	61	=	124	|
40	（	62	>	125	}
41	）	63	?	126	~
42	*	64	@	127	DEL(delete键)
43	+	65-90	A-Z		

字符串常量

1. 定义：在程序中用一**对双撇号把若干个字符**括起来，
   作为字符串常量（但不包括一对双撇号）。
   例：“123”，“boy”,“a”等。

   字符串到\0结束后面的不要了

   占用空间个数：是串长度串中字符个数加1

   串长度：实际字母的长度不算\0结束标志
2. 说明：
   (1)字符串常量只能用**双撇号**，不能用单引号括起来；如：‘A’不是字符串常量
   (2)字符串常量**不能参与算数运算。**
3. 字符常量和字符串常量区别
   - 分界符不同字符串常量双撇号，字符常量单撇号
   - 字符串常量内容多个字符，字符常量一个字符
   - 占用空间不同字符串常量存储结尾**\0** (占用空间字节个数是字符数加1因为\0也占用一个字节)
     字符常量占用1个字节
   - 字符参与运算，字符串不能参与运算

符号常量

1. 用**编译预处理宏定义**一个符号名的方法来代表一个常量
   格式：**#define 宏名 宏值**
2. ```
   #include <stdio.h>
   #define PI 3.14159
   main(){
       float r;
       double s;
       r=5.0;
       s=PI*r*r;
       printf("s=%f\n",s);
   }
   ```

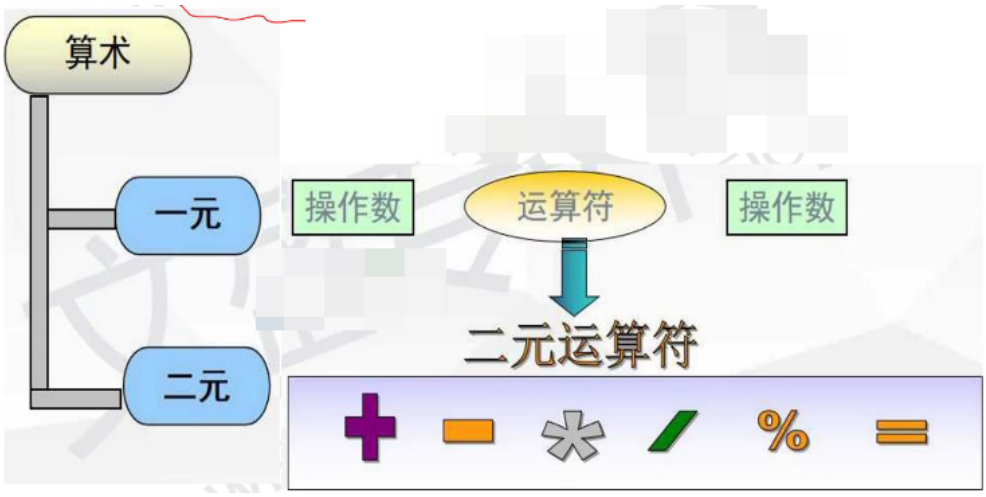
二、变量

1. 定义：在**程序运行过程中，值可以改变**的量
2. 说明
   - 每个**变量**有一个**名**字作为标识，属于用户标识符。
   - **变量**必须**先定义后使用**（定义后还得赋值才能用）
   - 变量代表了内存中的若干个存储单元，**变量名**实际上是以一个名字**代表**的一个**存储空间**（存储地址）
   - **变量名**和**变量值**是两个不同的概念
     - 定义变量时指定该变量的名字和类型
     - 从变量中取值，实际上是通过变量名找到相应的内存地址，从该存储单元中读取数据。
3. 声明变量：
   - 格式：**数据类型 变量名**；
     如：int i
4. 变量初始化：
   - (1)**边定义变初始化： 数据类型 变量名=值**； （注意区分变量名和变量值）
   - (2)**先定义后初始化：数据类型变量名**； 变量名=值；
5. 常用定义变量的数据类型：

   int(4个字节) char(1个字节) float(单精度4字节) double(双精度4字节) （长度和机器有关）

三、运算符

算术运算符

1. 
   ```

2. 基本的算术运算符：

+:正号运算符（单目运算符）
-:负号运算符（单目运算符）
*:乘法运算符
/:除法运算符
%:求余运算符
+:加法运算符
+:加法运算符
++:自增运算符
--:自减运算符

- 1、算术运算符
  - 1：* / % 双目/二元 从左向右 + - 双目 从左向右

    若以上5个运算符混合，* / % 高于+ -，但是小括号()改变优先级
  - 2：**%求余要求两个操作数都为整数**
  - 3：类型转换(自动类型转换)：
    - char char->int
    - int char->int
    - int int->int
    - char/int float/double--->double
    - float float/double--->double
    - double double--->double
    - 注意：商 取整 向0 取整（下取整）
  - 4：**++ -- 自增 自减 单目 需要一个操作数（整型）**
    - **前缀 ++i --i 先变后用新值 先自增1在用新值 i=i+1在用新i 减同理**
    - **后缀 i++ --i 先用原值后变 先用i i在自增1 先用 i 原值 i=i+1**
    - **注意：前缀和后缀只有在表达式才有区别若作为单独语句功能相同没有区别**
- 2.注意
  - (1)“%”的运算对象只能是整数，记法：整数%整数
  - (2)双目运算符运算结果类型
    - 运算数类型一致，结果类型同运算数类型。
    - **运算数类型不一致，**系统自动进行类型转换（**由低向高**转换)
      例：3/2的结果就是 1 3/2.0的结果就是1.5
  - (3)**强制类型转换：（类型名）(表达式)**
    如：3.5%2错误 （int）3.5%2正确

赋值运算符

1. 值运算符和赋值表达式
   变量名=表达式
2. ※说明：
   1.优先级倒数第二，结合方向：右——>左
   2.**是一种赋予的关系而不是等价的关系**
   3.=**左侧只能是变量**，不能是表达式

4.赋值号右边任意
可以是一个赋值表达式吗?

复合赋值表达式

1. 复合赋值运算符
   +=、-=、*=、/=、%=、《=、》=等（两个运算符之间不能有空格)
2. **借助复合的赋值运算符**将形如：
   “变量名=变量名+表达式”的表达式
   简化成：“**变量名+=表达式**”的形式

   说明：凡是有赋值运算符参加的运算都是从右往左算例：

   a+=3 等价于a=a+3
   x * =y+8 等价于x=x*(y+8)
   x%=3等价于x=x%3

逗号运算符和逗号表达式

1. 定义：用逗号运算符将表达式连接起来的式子
2. 一般形式：表达式1，表达式2，表达式3，...，表达式n
3. 求解过程：
   从左到右一个一个求解，**最后一个表达式的值**就是整个、逗号表达式的值。
4. 说明：（1)结合方向左--右　　(2)**优先级最低**
   例如：已知a=3　a=3,a+3结果：表达式的值为6
   　　b=a+3,a-3结果：表达式的值为0
   问：a=(2,3,4)a的值是多少？ a=4
   　　a=2,3,4 a的值是多少？　a=2 后面不保留

c语言运算优先级

- 一个表达式可以包含多个运算符。在这种情况下，运算符的优先级决定表达式的哪部分被处理为每个运算符的操作数。例如，按照运算规则，表达式中 *、/、以及 % 的优先级比 + 和 - 高。如下列表达式：

  ```
  a — b * c
  ```

  相当于 a - （b*c）。如果想让操作数以不同的方式组合在一起，则必须使用括号：

  ```
  (a — b) * c
  ```

- 如果一个表达式中的两个操作数具有相同的优先级，那么它们的结合律（associativity）决定它们的组合方式是从左到右或是从右到左。例如，算术运算符和操作数的组合方式是从左到右，赋值运算符则是从右到左，如表 1 所示。

表达式	结合律	组合方式
a/b%c	从左到右	（a/b）%c
a=b=c	从右到左	a=（b=c)

- 表 2 列出优先级次序下，所有 C 语言运算符的优先级和结合律。

优先级	运算符	结合律
1	后缀运算符：[] () · -> ++ --(类型名称){列表}	从左到右
2	一元运算符：++ -- ！ ~ + - * & sizeof_Alignof	从右到左
3	类型转换运算符：(类型名称)	从右到左
4	乘除法运算符：* / %	从左到右
5	加减法运算符：+ -	从左到右
6	移位运算符：<< >>	从左到右
7	关系运算符：<<= >>=	从左到右
8	相等运算符：== !=	从左到右
9	位运算符 AND：&	从左到右
10	位运算符 XOR：^	从左到右
11	位运算符 OR：|	从左到右
12	逻辑运算符 AND：&&	从左到右
13	逻辑运算符 OR：||	从左到右
14	条件运算符：?:	从右到左
15	赋值运算符： = += -= *= /= %= &= ^= |= <<= >>=	从右到左
16	逗号运算符：，	从左到右
- 表 2 中优先级最高的运算符中的最后一个（类型名称）{列表} 是 C99 新增加的。
- 一些运算符记号在表 2 中出现了两次。例如，自增运算符 ++ 和自减运算符 --，在作后缀运算符（如表达式 x++）时，较其用作前缀运算符（如表达式 ++x）时，具有较高的优先级。
- +、-、* 和 & 运算符记号不但可以当作一元运算符（unary operator，只需要一个操作数），也可以当作二元运算符（binary operator，需要两个操作数）。例如，* 只有一个操作数的时候，就是间接运算符（indirection operator），而有两个操作数的时候，就是乘号。

- 在这些例子中，一元运算符比二元运算符具有更高的优先级。例如，表达式 *ptrl**ptr2* 等同于表达式（ptrl）（ptr2）。

第二章：流程控制结构

顺序结构

语句

- 语句：C语言中描述计算过程的最基本单位。 由分号；结束。
- 顺序结构：按语句在程序中出现的顺序逐条执行，没有分支、没有转移。

复合语句和空语句

1. 复合语句
   - 定义：用一对**花括号**把若干语句括起来构成一个语句组。
   - 注意：
     (1)花括号内语句的数目不限
     (2)在**花括号外面不能加分号**
2. 空语句

```
main(){
 ;
}
```

数据输出

1. 基本概念
   1. 输出：把数据从计算机**内部**送到计算机**外部设备**上的操作称为"输出”
   2. 输入：从计算机**外部设备**将数据送入**计算机内部**的操作称为”输入”。
2. 注意：C语言本身不提供输入和输出语句，但是有输入和输出函数。
3. 在使用输入输出函数时，要在程序文件的开头用预编译指令

```
#include <stdio.h> /根目录
或
#include "stdio,h" /用户目录
```

4. printf()函数
   1. 作用：在终端设备上按指定格式输出
   2. 一般调用形式：

      表达式：printf (格式控制，输出项表)
      语句：printf (格式控制，输出项表);
   3.

格式控制	说　明
%d	将参数按整数形式转换输出，对应参数应是int类型。
%ld	将参数按长整数形式转换输出，对应参数应是long类型。
%c	输出一个字符，对应参数应该是一个字符。
%s	输出一个字符串，对应参数应该是一个字符串。
%f	将参数按带小数点数形式输出，默认情况下精确到6位小数。

   4. 常用格式字符

1. d格式符：用来**输出一个有符号的十进制整数**
   - 指定列宽：**%md** 右对齐 **%-md** 左对齐

     printf(”%5d%-5d\n”,12,-345);

     输入的数不够m先填几个空列右对齐前面左对齐后面
   - %d输出int型数据
   - %ld输出long型数据
2. c格式符：用来输出一个字符

```
char ch='a';
printf("%c",ch;
printf("%5c",ch);
```

   输出字符：a
3. s格式符：用来输出一个字符串

```
printf("%s","CHNA")
```

   输出字符串：CHINA
4. f格式符：用来输出实数，以小数形式输出
   - 下指定数据宽度和小数位数，用%f

     例：用%输出实数，只能得到6位小数。

```
double a=1.0;
printf("%f\n",a/3);
0.333333
```

   - 指定数据宽度和小数位数：用%m.nf(右对齐) m列宽 n小数位数

```
printf("%20.15f\n",1.0/3);
//0.333333333333333 15位小数
printf("%.0f\n",10000/3.0):
//3333
float n=9.478689;
printf("%f",n);
//输出结果：9.47869
//默认情况下精确到六位小数 上面是7位四舍五入
//m宽度，表示所有的数字和小数点所占的位数，不够20位右对齐。
//n精度（精确到小数点后多少位）
```

   - 输出的数据向左对齐，用%-m.nf
5. **e(E)**格式符。指定以**指数形式输出**实数
   - %e,VC++给出小数位数为6
   - 小数点前必须有而且只有1位非零数字

```
printf("%e",123.456;
//输出：1.234560 e+002  002表示10的2次方
```

- %m.ne 设置列宽和小数位数 m列宽 n小数位数

```
printf("%13.2e",123.456)
//输出：     1.32 e+001 前面4个空格
```

5. 注意
   - 遇到%号字符，按后面输出列表变量的值代替
   - 遇到 \ 转义符体现功能
   - 其他一般字符原样输出
   - 字符型用%d输出是字符码的值
6. 注意事项
   1. 格式控制中应包含与输出项**一一对应**的输出格式说明，类型必须匹配；
      (1)若格式说明的**个数少于输出项个数**，则多余的输出项不予输出。

      ```
      如：int x=12,y=28;printf('%d',x,y);
      ```

      (2)若格式**说明的个数多于输出项个数**，输出乱码。

      ```
      //如：
      int x=12,y=28;
      printf("%d%d%d",x,y);
      ```

   2. **同一变量，不同形式**，出现在同一条输出函数调用中。

      ```
      //如：
      int k=21;
      printf("%d,%d",k,++k);
      //输出22,22
      ```

      原因：printf函数其参数**从右往左进行处理**，先计算++k。显示值时，从左往右。其实k是先执行的但是进入栈了++k后进栈 但是栈后进先出

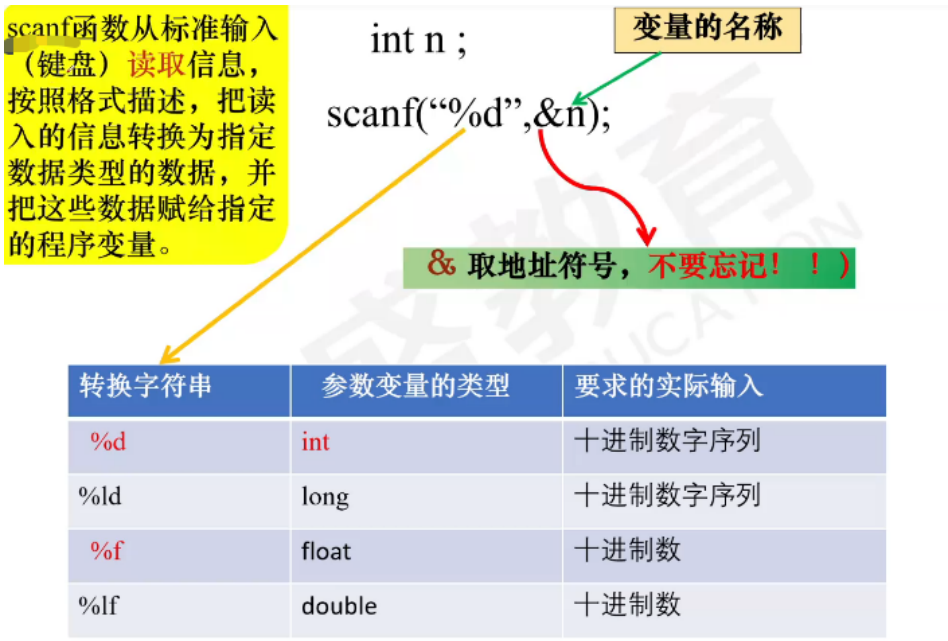
数据输入

1. scanf函数的一般调用形式
   - 作用：在终端设备上输入数据
   - 一般调用形式：
     表达式：scanf(格式控制，地址表列)
     语句：scanf(格式控制，地址表列);

     格式控制含义同printf函数
     地址表列可以是变量的地址，或字符串的首地址
   - 注意：格式控制必须与你对应的变量的类型相等，否则会出现意想不到的数据。
     记忆：第一部分格式控制的形式在终端输入数据　一模一样！
   - scanf函数中的格式声明

     %+格式字符，中间可以插入附加的字符，**附加字符必须原样输入。**

```
scanf("a=%f,b=%f,c=%f",&a,&b,&c);
//a=是附加字符输入的时候必须输入a=...
```

- 

转换字符串	参数变量的类型	要求的实际输入
%d	int	十进制数字序列
%ld	long	十进制数字序列
%f	float	十进制数
%lf	double	十进制数

2. 使用scanf函数时应注意的问题

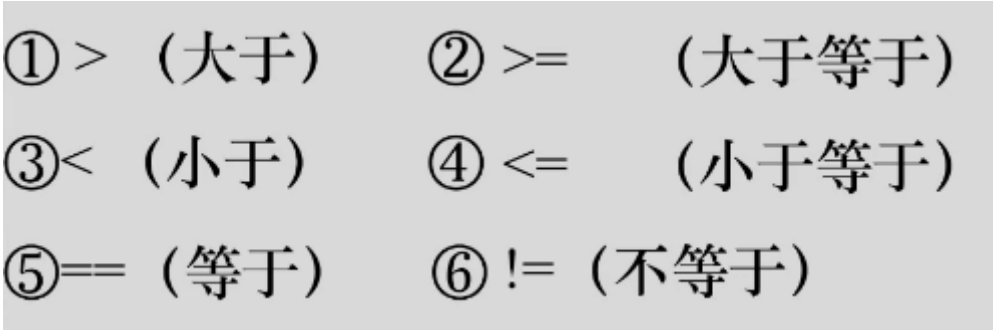
- ```
  scanf("%f%f%f",a,b,c);//错
  scanf("%f%f%f",&a,&b,&c);//对
  scanf("a=%f,b=%f,c=%f",&a,&b,&c);
  //132错
  //a=1,b=3,c=2对
  //a=1b=3c=2错
  scanf("%c%c%c",&c1,&c2,&c3);
  //abc对
  //a b c 错
  对于scanf("%d%c%f",&a,&b,&c);
  //若输入1234a123o.262
   a=1234 b=a c=123 剩下不要但不是不存在
  ```

选择结构

C语言中的逻辑值

1. 逻辑值只有两个，分别用“真”和“假”表示。
2. **任何基本类型的值都可作为逻辑值**使用。
   C语言没有专门的逻辑值，**所有非0**的值被都被当作“真使用，而0值被当作“假”使用。
3. 关系运算符、关系表达式
   - 关系运算实际上是“比较运算”
  ```

- C语言的关系运算符共6种：



- **优先级**：前四种优先级高于后两种
  结合方法：从左向右
- 关系运算符的操作数可以是**变量、常量或表达式。**
- 关系表达式的
  计算结果=逻辑值（真或假）
  在C语言中，“0”表示“假”，“非0”表示“真”

4. 逻辑运算符、逻辑表达式

- 3种逻辑运算符：
  &&(逻辑与) ||(逻辑或) !(逻辑非)
- 双目 || && 单目 !
- 逻辑运算符的优先次序

  ! → && →|| (!为三者中最高)
- 逻辑运算符与逻辑表达式

  非取反真变假假变真

  逻辑与同真为真其他为假

  逻辑或同假为假其他未真

5. 运算符的优先次序

- 运算次序：
  ！ （高）
  算术运算符
  关系运算符
  &&
  ||
  赋值运算符 （低）

6. 逻辑运算符与逻辑表达式

- 表示逻辑运算结果时：1“真”，0“假”
- 判断一个量是否为“真”时：0“假”，非0“真”
- 注意：将一个非零的数值认作为“真”
- 短路

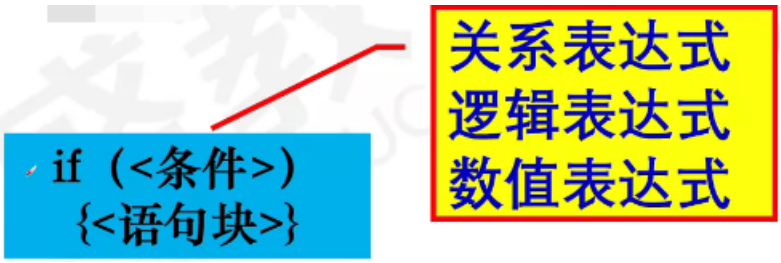
  与遇见0就**短路 后面不算**结果0

  或遇见1就 不算结果1

选择结构

1. 选择结构：根据条件进行判断真假，执行不同的操作。

简单if语句的一般形式为

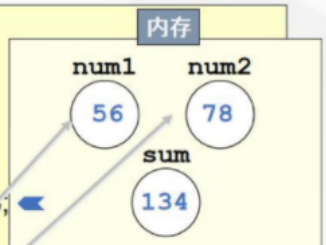


- 如果条件为真，语句执行一个语句或一组语句；
- 如果条件为假，则执行if语句后面(else)的语句（如果有)。
- if-else语句的一般形式为：

```
if(<表达式){
<语句块>
}else{
<语句块>
}
```

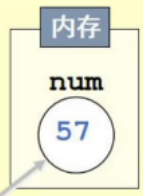
-

```
# include <stdio.h>
void main()
{
  int  num1, num2, sum;
  printf("\n请输入两个数：");
  scanf("%d %d", &num1,&num2);
  sum = num1+ num2 ;
  if(sum > 100)
        printf("\n两数的和大于 100 \n ");
}
```




```
请输入两个数：56 78
两数的和大于 100
```

```
#include <stdio.h>
void main()
{
        int num;
        printf("\n请输入一个整数：");
        scanf ("%d", &num);
        if((num % 2) == 0)
                printf("%d 是一个偶数。\n", num);
        else
                printf("%d 是一个奇数。\n", num);
}
```




```
请输入一个整数：57
57 是一个奇数。
```

```
#include <stdio.h>
void main()
{
  int  num;
  printf("\n 请输入一个数：");
  scanf("%d", &num);
  if (!(num %5))
        printf("\n 该数能被 5 整除 \n ");
  else
        printf("\n 该数不能被 5 整除 \n ");
}
```

```
请输入一个数：90
该数能被 5 整除
```

```
#include <stdio.h>
void main()
{
  int year;
  printf("\n 请输入年份：");
  scanf("%d",&year);
  if((year % 4 ==0 && year % 100 != 0) || (year % 400 == 0))
        printf("\n %d 年是闰年 \n ", year);
  else
        printf("\n %d 年不是闰年 \n ", year);
}
```

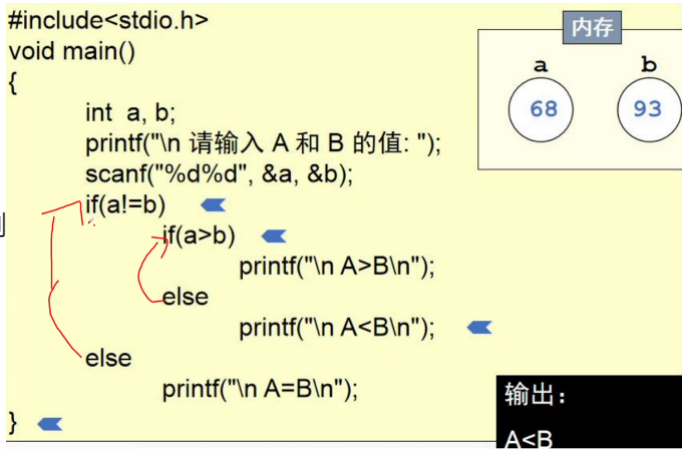
```
请输入年份：2005
2005 年不是闰年
```

闰年判定：年份能被4整除**但**(&&)不能100整除，**或者**(||)年份能被400整除

2. 嵌套if

匹配规则

- 有else必有if
- 匹配：else与它前面的、紧挨着的、未被匹配的相匹配。
- 例



- if下面没括号是1行其他行是正常语句

3. 条件运算符和条件表达式

- 条件表达式的一般形式为：
  表达式1？表达式2：表达式3
- 条件运算符的执行顺序：

  ◆求解表达式1
  ◆真：求解表达式2，值为整个条件表达式的值
  ◆假：求解表达式3，值是整个条件表达式的值

  注：唯一的三目运算符 记忆： 真前假后
- 算术>关系>逻辑>条件>赋值>逗号
- 结合性：“自右至左”
- 以下为合法的使用方法：

```
a>b?(max=a):(max=b);
a>b?printf("%d",a):printf"%d",b)
```

-

```
#include <stdio.h>
void main()
{
    double sal;
    double rate;
    printf("\n请输入基本工资:");
    scanf("%lf",&sal);
    rate = (sal<=1000) ? 0 : 0.05;
    sal = sal-(sal-1000)*rate;
    printf("\n 税后工资为:%7.2f\n", sal);
}
```

```
请输入基本工资： 1500
税后工资为： 1475.00
```

第三章：函数

第四章：数组

第五章：指针

第六章：结构体和文件

第七章：综合练习