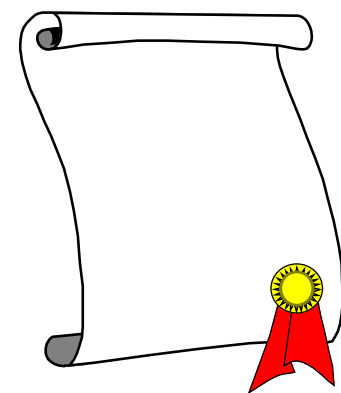


第三章

最简单的C程序设计 ——顺序程序设计

主要内容

- 一、C的数据类型
- 二、常量与变量
- 三、基本数据类型
- 四、运算符与表达式



一、C的数据类型

1、什么是数据类型？

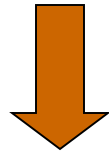
2、C语言的数据类型有哪些？

1、什么是数据类型？

对数据的描述

对操作的描述

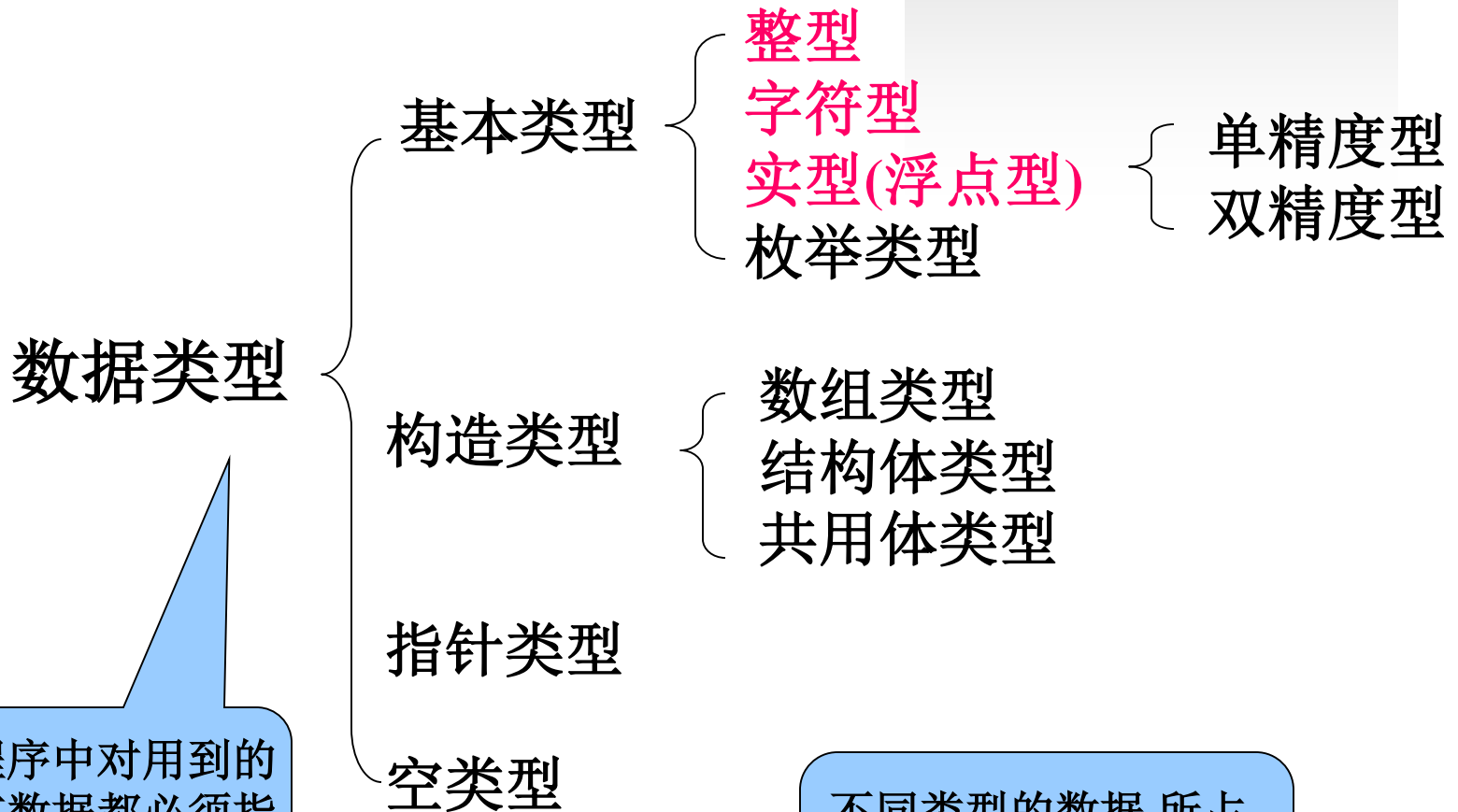
数据结构+算法=程序



“数据类型”

是构造数据结构的基础

2、C的数据类型



在程序中对用到的所有数据都必须指定其数据类型。

不同类型的数据,所占内存的字节数不同,对应的操作也不相同。

二、常量与变量

1、 常 量

2、 变 量

1、常量

概念

在程序运行过程中，其值不能被改变的量。

字符串常量

分类

直接常量： 12 -44.6 'a' "Hello! "

符号常量： 用一个**标识符**代表一个常量。

符号常量的定义

格式： #define 常量名 直接常量

举例：

```
#define PRICE 100
#define PI 3.14
#define C1 'a'
```

习惯上常量名用大写，
变量名用小写。

符号常量的定义放在
函数外面。

意义

- 1、含义清楚。
- 2、在需要改变一个常量时，能够做到“一改全改”。

printf进阶——输出常量

- 复习:
- `printf(“这里是你要输出的内容。\\n”);`

printf进阶——输出常量



- `printf(“格式控制”,常量);`
- `printf(“%d”,123);`//整型
- `printf(“%f”,1.23);`//浮点型
- `printf(“%c”,’a’);`//字符型
- `printf(“%s”,”hello”);`//字符串

练习

利用printf格式控制%d和%s输出：

15和15的差是： 0

符号常量——举例

```
#include <stdio.h>
#define PRICE 100

int main()
{
    printf("%d\n",PRICE);
    return 0;
}
```

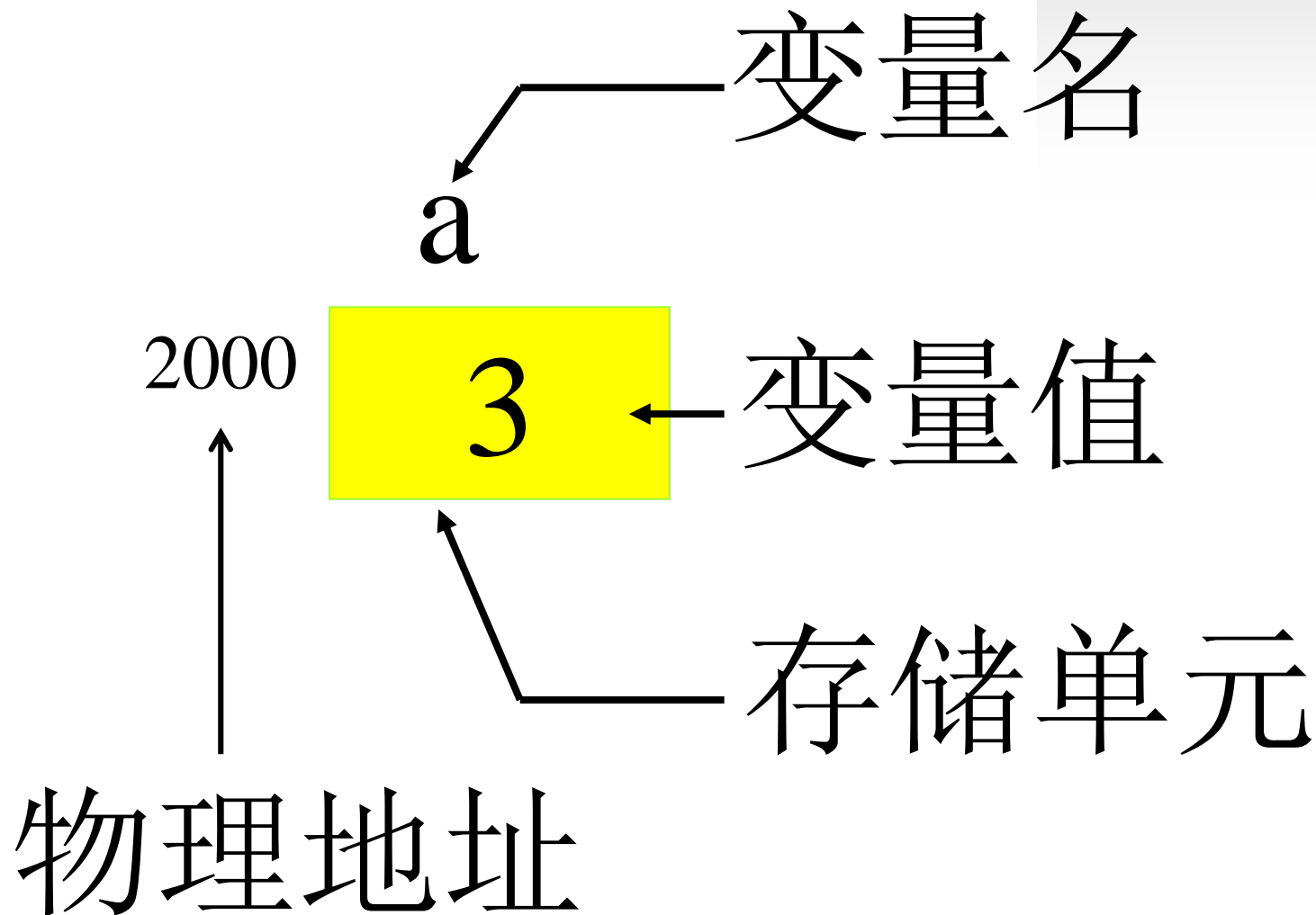
2、变量

概念 其值可以改变的量称为变量。

要素

- (1) **变量名**。每个变量都必须有一个名字——变量名，变量命名遵循标识符命名规则。代表内存中的一个存储单元。
- (2) **变量值**。在程序运行过程中，变量值存储在内存中。在程序中，通过变量名来引用变量的值。

变量图示



标识符
identifier

- 1、标识符就是一个名字（如**常量名、变量名、函数名**……）。
- 2、组成：只能由**字母、数字、下划线**组成。且第一个字符必须为**字母或下划线**。
- 3、长度：标识符的有效长度随系统而异，如果超长，则超长部分被舍弃。VC++6.0中规定最大长度为247个字符。
- 4、标识符命名的良好习惯——见名知意
例如，**name**（姓名）、**age**（年龄）

思考？

下列标识符是否合法，为什么？

① sum ② a.b.c ③ b2 ④ a1 ⑤ 6a8bc

⑥ s_name ⑦ #ss ⑧ M.D.John

⑨ Tom&Jerry ⑩ a>b

变量定义 数据类型 变量名[, 变量名2.....];

int	i;
int	i1,i2;
float	f1,f2,f3;
double	d1,d2;
char	c1,c2;

习惯上，变量名用小写字母。
常量名用大写字母。

变量赋初值

出现在函数体的
声明部分

方式:

1、定义变量的同时赋初值（也叫变量初始化）。

格式:

数据类型 变量名[=初值][, 变量名2[=初值2].....];

2、通过一个赋值语句给变量赋初值。

格式:

变量名=初值;

出现在函数体的
执行部分

```
int a,b=3;
```



```
int a,b;  
b=3;
```

变量赋初值

```
sales = 4432.67 / 1.20;
```

=

3693.8916

名为sales
的变量

图5-1 赋值运算符把值放进变量中

关于变量以及变量赋初值的几点说明

- 1、程序中用到的变量必须“先定义（declare），后使用”。
- 2、C语言的**关键字**不能用作变量名。
- 3、C语言对英文字母的**大小写敏感**，即同一字母的大小写，被认为是两个不同的字符。
- 4、定义变量时，给几个变量赋相同的初值，
应写成：`int a=3,b=3,c=3;`
不能写成：`int a=b=c=3;`
- 5、给变量赋值时，正常情况下应给变量赋相同类型的数据。若给变量赋与其类型不同的数据时，需进行**类型转换**(后面介绍赋值语句时详细介绍)。

变量——举例

本程序中有
几处错误？

```
#include <stdio.h>

int main()
{
    int student,age;
    int if=adrrress=1;
    float score=90;

    staden=2;
    Age=20.7;
    printf("%d %d %d %f",if,student,age,score);
    return 0;
}
```

undeclared identifier

三、基本数据类型

1、整数类型

2、实数类型

3、字符类型

4、不同类型数据间的转换

1、整数类型——变量

各类型整型变量占用的内存字节数，随系统而异。

分类及取值范围

	类型	比特数	范围
int	[signed] int	16	- 32768~32767
	unsigned int	16	0~65535
short [int]	[signed] short [int]	16	-32768~32767($2^{15}-1$)
	unsigned short [int]	16	0~65535
long [int]	[signed] long [int]	32	-2147483648~ 2147483647
	unsigned long [int]	32	0~4294967295

[]表示可以省略

定义整型变量并赋值


- `int zhengxing=5;`
- `short int duanzhengxing=5;`
- `long int changzhengxing=5;`
- 有区别吗？

整数类型——溢出举例

```
#include <stdio.h>

int main()
{
    int a,b;
    a=2147483647;
    b=a+1;
    printf("a = %d\nb = %d",a,b);
    return 0;
}
```

运行结果



```
a = 2147483647
b = -2147483648
```


1、整数类型——常量

表示方法

{ 十进制: 123
 八进制: 0123, -0123
 十六进制: 0x123, -0x4

思考：以下对整型常量的书写方法是否合理？

0X10F, -o123, 0482, 900

如123l, 315L

分类

{ 基本整型（数据范围与int型一样）
 长整型（在数值后面加“L（l）”，
 数据范围与long int型一样）

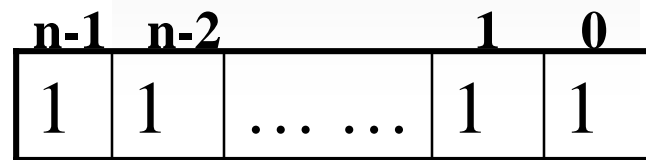
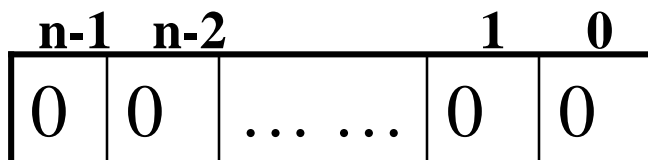
类型匹配规则

一个整型常量，可以赋给能容纳下其值的整型变量

各个类型“可能”占据的空间

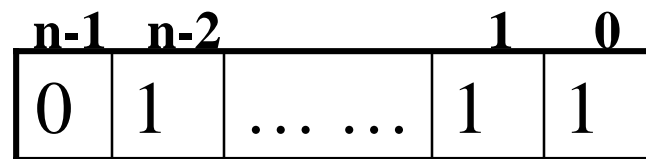
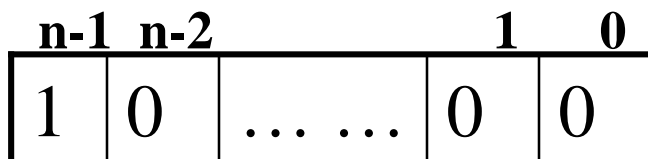
类型	占据空间
char, unsigned char, signed char	1 byte
short, unsigned short	2 bytes
int, unsigned int	4 bytes
long, unsigned long	4 bytes
float	4 bytes
double	8 bytes
long double	8 bytes

n位无符号整数的取值范围:

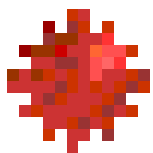


即 $0 \sim 2^n - 1$

n位带符号整数的取值范围:



即 $-2^{n-1} \sim 2^{n-1} - 1$



注意：

常量无**unsigned**型。但一个非负整型常量，只要它的值不超过相应变量的值域（即取值范围），也可以赋给**unsigned**型变量。

举例

```
#include <stdio.h>
void main()
{
    long int c,d=9L;
    unsigned int e=-3, f;
    f =5;
    c=-6;
    printf("%ld\n%ld\n%u\n%u",c,d,e,f);
}
```

运行结果



```
-6
9
4294967293
5
```

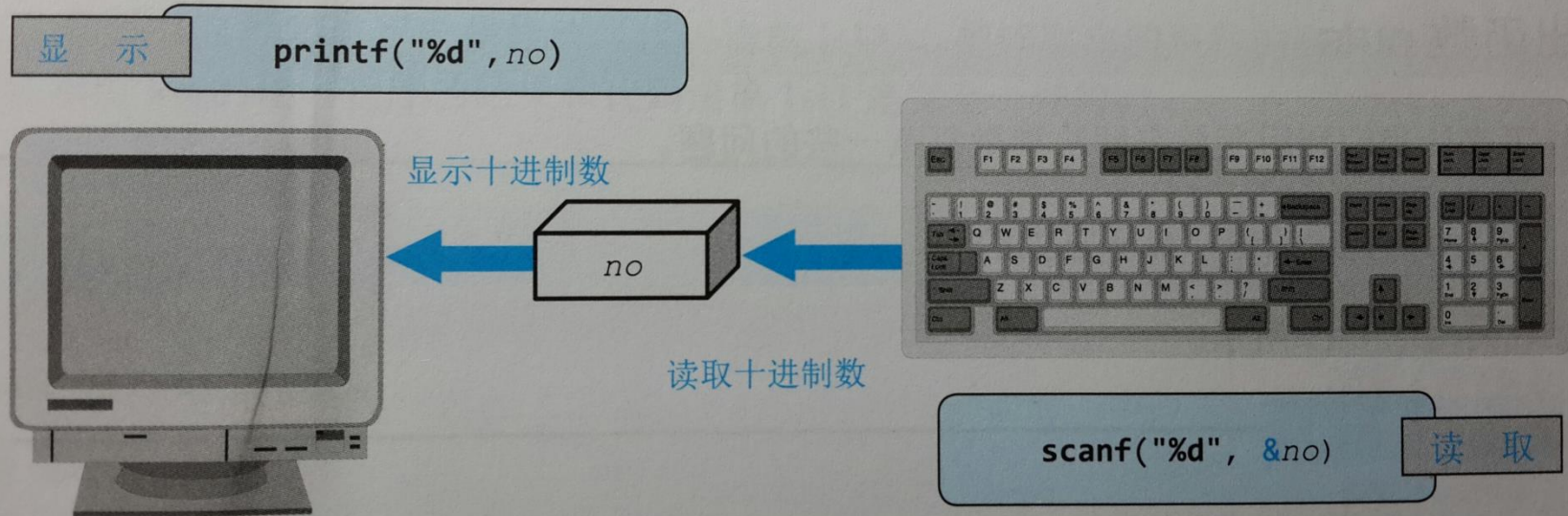
整数的输入和输出

输入： 将从键盘输入的整数存储到a中
`scanf("%d",&a);`

输出： 将a的值输出到屏幕
`printf("%d",a);`

输入和显示

输入和显示



示例程序

```
#include<stdio.h>  
int main()  
{  
    int no;  
    scanf(“%d”,&no);  
    printf(“%d”,no);  
    return 0;  
}
```

示例程序-人性化

```
#include<stdio.h>

int main()
{
    int no;
    printf("请输入一个整数: ");
    scanf("%d",&no);
    printf("您输入的整数是: ");
    printf("%d",no);
    return 0;
}
```


练习

- 输入两个数
- 输出这两个数的和 【差、积、商】
- 提示：
 - $\text{sum} = a + b$
 - $\text{sum} = a - b$
 - $\text{sum} = a * b$
 - $\text{sum} = a / b$

做四个程序或者一个程序均可

应用举例

- 例**2.1** 鸡兔同笼。在一个笼子里同时养着一些鸡和兔子，你想了解有多少只鸡和多少只兔，主人对你说：我只告诉你鸡和兔的总头数是**16**和总脚数是**40**，你能不能自己计算有多少只鸡和多少只兔？

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    int heads,foots;
```

```
    int rabbits,chicken;
```

```
    heads=16;
```

```
    foots=40;
```

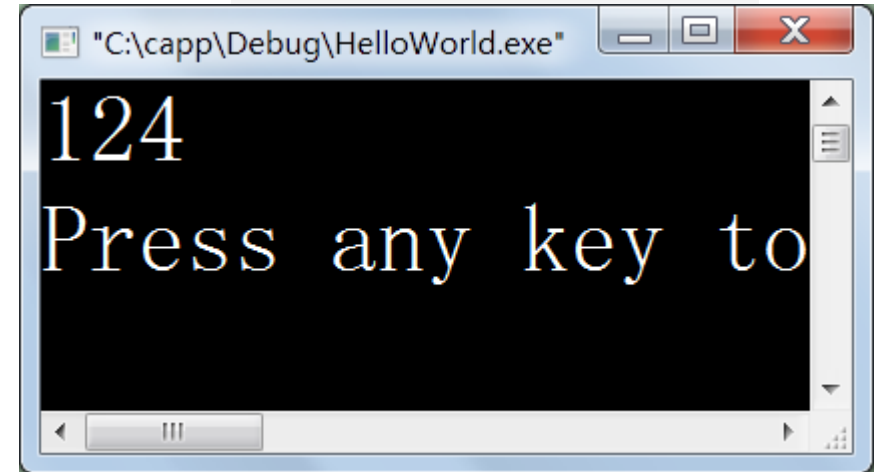
```
    rabbits=(foots-2*heads)/2;
```

```
    chicken=heads-rabbits;
```

```
    printf("%d%d\n",chicken,rabbits);
```

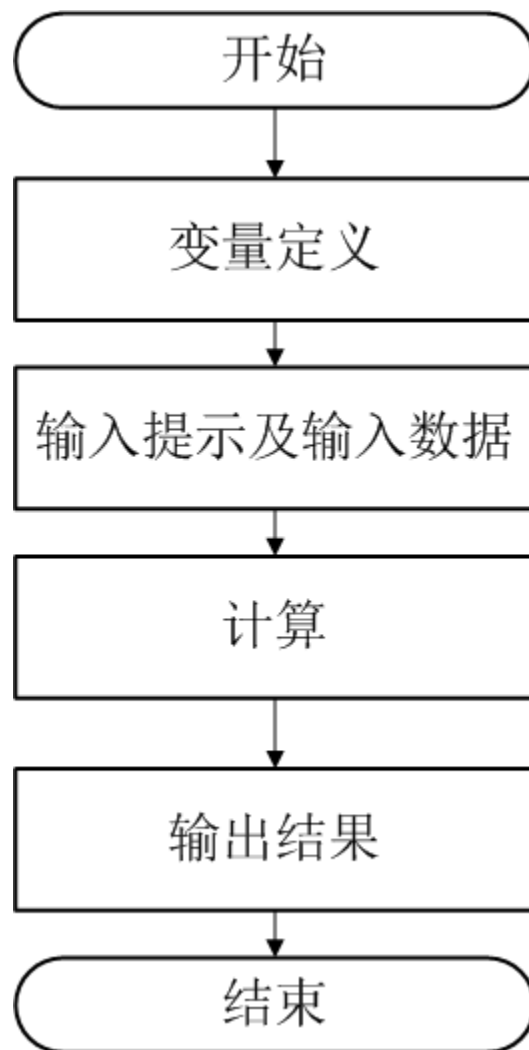
```
    return 0;
```

```
}
```



```
#include<stdio.h>
int main()
{
    int heads,foots;
    int rabbits,chicken;
    printf("请输入头的数量: ");
    scanf("%d",&heads);
    printf("请输入脚的数量: ");
    scanf("%d",&foots);
    rabbits=(foots-2*heads)/2;
    chicken=heads-rabbits;
    printf("鸡的数量为: %d\n",chicken);
    printf("兔的数量为: %d\n",rabbits);
    return 0;
}
```

用流程图描述写程序的“八股”



2、实数类型——变量

分类及取值范围

类型	比特数	有效数字	数值范围
float	32	6~7	$-10^{-37} \sim 10^{38}$
double	64	15~16	$-10^{-307} \sim 10^{308}$
long double	128	18~19	$-10^{-4932} \sim 10^{4932}$

2、实数类型——常量

数字和小数点组成

表示方法

{ 十进制小数形式: 123.34, -0.0045, 0.0
指数形式: <尾数>E (e) <整型指数>
如0.23e2 , -0.45e-3, **3.0 E +5**

规范化指数形式

关于分类

C编译系统将实型常量作为**双精度**来处理。
一个实型常量，可以赋给一个实型变量
(float型或double型)。

实型变量的输入与输出

输入:

`scanf("%f",&a);`//适用于float类型

`scanf("%lf",&a);`//适用于double类型

输出:

`printf("%f",a);`

2.3.1 实型数据的运算举例

例2.2 银行存款定存1年，利率3.25，若存入10000元，到期自动转存，仍为1年定期，5年后得到多少钱。

```
#include <stdio.h>

int main()
{
    int qian=10000;
    float lilv=0.0325;
    qian=qian* (1+lilv) ;
    printf("qian=%f\n",qian);
    return 0;
}
```

```
#include <stdio.h>

int main()
{
    int qian=10000;
    float lilv=0.0325;
    qian=qian* (1+lilv) ;
    printf("qian=%f\n",qian);
    return 0;
}
```

改进代码，
完成要求功能，
并增加用户
可理解性，
可操作性。

```
#include <stdio.h>

int main()
{
    int qian=10000;
    float lilv=0.0325;
    qian=qian* (1+lilv) ;
    printf("qian=%f\n",qian);
    return 0;
}
```

改进代码，
完成要求功能，
并增加用户
可理解性，
可操作性。

例2.2 银行存款定存1年，利率3.25，若存入10000元，到期自动转存，仍为1年定期，5年后得到多少钱。

小结：实型变量的输入与输出 与整型变量的输入与输出

输入： `scanf("%f",&a);`

输出： `printf("%f",a);`

输入： `scanf("%d",&a);`

输出： `printf("%d",a);`

练习

- P84
- 设半径 $r=1.5$ ，圆柱高 $h=3$ ，求圆周长、圆面积、圆球表面积、圆球体积、圆柱体积。
用scanf输入数据，输出计算结果。
- 输出时要有文字说明
- 输出时保留小数点后2位数字

3、字符类型——常量

不同的两个字符

表示方法

用单引号括起来的一个字符. 'a' 'A' '9' '=' '?'

转义字符

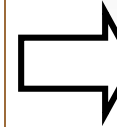
C语言还允许使用一种特殊形式的字符常量，就是以反斜杠 '\ ' 开头的转义字符。

将\后面的字符转换为另外的意义.即是一种控制符号.

字符形式	含义 (P48)
\n	换行,从当前位置移到下一行开头
\t	水平制表(跳到下一个tab位置)
\b	退格,将当前位置移到前一列
\r	回车,将当前位置移到本行开头
\'	单撇号字符
\\	反斜杠字符
\"	双撇号字符
\ddd	1~3位8进制数所代表的字符
\xhh	1~2位16进制数所代表的字符
.....

举例

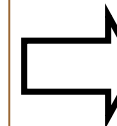
```
#include <stdio.h>
int main()
{
    printf("abc\bd\nxyz");
    printf("\rlmn\topq\n");
    return 0;
}
```



结果:

```
abd
lmn  opq
```

```
#include <stdio.h>
int main()
{
    printf("\x4F\x4B\x21\n");
    printf("\x15 \xAB\n");
    return 0;
}
```



结果:

```
O K !
§  ½
```


字符类型——变量

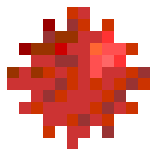
类型及范围 **char** 大小写英文字母,数字,运算符,标点符号等

存储方式 以ASCII码存储,占一个字节

举例 字符变量**ch**中存放字符'a', 其ASCII码为97, 在内存中以二进制形式存放, 其格式如下:

ch

0 1 1 0 0 0 0 1



注意: 字符型数据与整型数据在存储方式上的相似性使得两者之间可以通用。但字符型数据的表示范围是: **0~255**。

字符类型的输入和输出

- 定义 : `char ch;`
- 输入 : `scanf("%c",&ch);`
- 输出 : `printf("%c",ch);`

字符类型——举例1

一个字符型数据，既可以字符形式输出，也可以整数形式输出

[例] 字符变量的字符形式输出和整数形式输出。

```
#include <stdio.h>
```

```
int main()  
{
```

```
    char ch1;  
    ch1='a';
```

```
    printf("ch1=%c\n",ch1);  
    printf("ch1=%d\n",ch1);
```

```
    return 0;
```

```
}
```

字符类型——举例2

允许对字符数据进行算术运算，此时就是对它们的**ASCII**码值进行算术运算

[例]字符数据的算术运算

```
#include <stdio.h>
int main()
{
    char ch1,ch2;
    ch1='a'; ch2='A';

    printf('ch1-32=%c\n',ch1-32);
    printf("ch2+32=%c\n",ch2+32);

    printf('ch1+200=%d\n', ch1+200);
    return 0;
}
```

程序运行结果：

字符类型

- 编写一个程序，将输入的小写字母变成大写字母
- `scanf("%c",&ch);`
- `printf("%c",ch);`

字符串常量

概念

用一对双引号" "括起来的若干字符序列

字符串 长度

字符串中字符的个数。长度为0的字符串（即一个字符都没有的字符串）称为空串，表示为""（一对紧连的双引号）。

存储方式

C语言规定：在存储字符串常量时，由系统
在字符串的末尾自动加一个'\0'作为字符串
的结束标志。

思考？

1、字符串常量"How do you do."和"Good morning."的长度分别是多少？

14和13

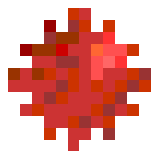
2、字符串C:\msdos\v6.22和I say: " Goodbye! "在C语言中的字符串常量形式如何表示？

"C:\\msdos\\v6.22"和"I say:\\\"Goodbye!\\\""

3、字符串"CHINA"在内存中的实际存储形式是怎样的？占用多少个字节的内存空间？

C	H	I	N	A	\0
---	---	---	---	---	----

占用 6 字节
而非 5 字节
内存空间



注意：



- 1、 'a'是字符， "a"是字符串。
- 2、 不能把一个字符串赋给一个字符变量。如char c; 那么c="a"、 c="Hello! "都是错误的。
- 3、 C语言中没有字符串变量，可以用字符数组来表示。

总结：字符与字符串的区别有哪些？

- ▶ 定界符不同：字符常量使用单引号，而字符串常量使用双引号；
- ▶ 长度不同：字符常量的长度固定为1，而字符串常量的长度，可以是0，也可以是某个整数；
- ▶ 存储要求不同：字符常量存储的是字符的ASCII码值，而字符串常量，除了要存储有效的字符的ASCII码值外，还要存储一个结束标志‘\0’。

基本数据类型总结

		变量	常量
整型	短整型	short [int]	12,012,0x12
	基本整型	int	
	长整型	long [int]	123L,123l
浮点型（实型）	单精度	float	12.34
	双精度	double	0.12e+2
	长双精度	long double	0.12E-2
字符型		char	‘a’, ‘\n’, ‘\2’, ‘\x2’
字符串		没有	“Hello!”

四、运算符与表达式

- (一) C运算符简介
- (二) 算术运算符和算术表达式
- (三) 赋值运算符和赋值表达式
- (四) 逗号运算符和逗号表达式
- (五) 关系运算符和关系表达式
- (六) 逻辑运算符和逻辑表达式

(一)、C运算符简介

分类	运算符
1、算术运算符	+ - * / %
2、关系运算符	> < == >= <= !=
3、逻辑运算符	! &&
4、位运算符	<< >> ~ ^ &
5、赋值运算符	=及其扩展赋值运算符
6、条件运算符	? :
7、逗号运算符	,
8、指针运算符	* &
9、求字节数运算符	sizeof
10、强制类型转换运算符	(类型)
11、分量运算符	. →
12、下标运算符	[]
13、其他	如函数调用运算符 ()

(二)、算术运算符和算术表达式

模运算符或求余运算符

基本算术运算符

+ - * / %

说明

两个整数相除的结果为整数，舍去小数部分。 例： $5/3=1$
当商为负数时，多采用“向零取整”的方法。 $-5/3=-1$

求余运算的两侧均应为整数 例： $5\%3=2$

算术表达式

用算术运算符和括号将运算对象（也称操作数）连接起来的、符合C语法规则的式子。

常量、变量、函数等

例： $a*b/c-1.5+'a'$

优先级和结合性

先乘除、后加减，有括号先算括号。
自左至右

说明

如果一个运算符两侧的数据类型不同，先自动进行类型转换，使二者具有同一类型，然后进行运算。

自增、自减运算符

作用：自增运算符（++）使单个变量的值增1；
自减运算符（--）使单个变量的值减1。

格式：前置：++i, --i （在使用i之前，使i的值加（减）1）
后置：i++, i-- （在使用i之后，使i的值加（减）1）

i=i+1

i=i-1

例 1

i=3;

.....

A) j=++i;

B) j=i++;

C) j=-i++;

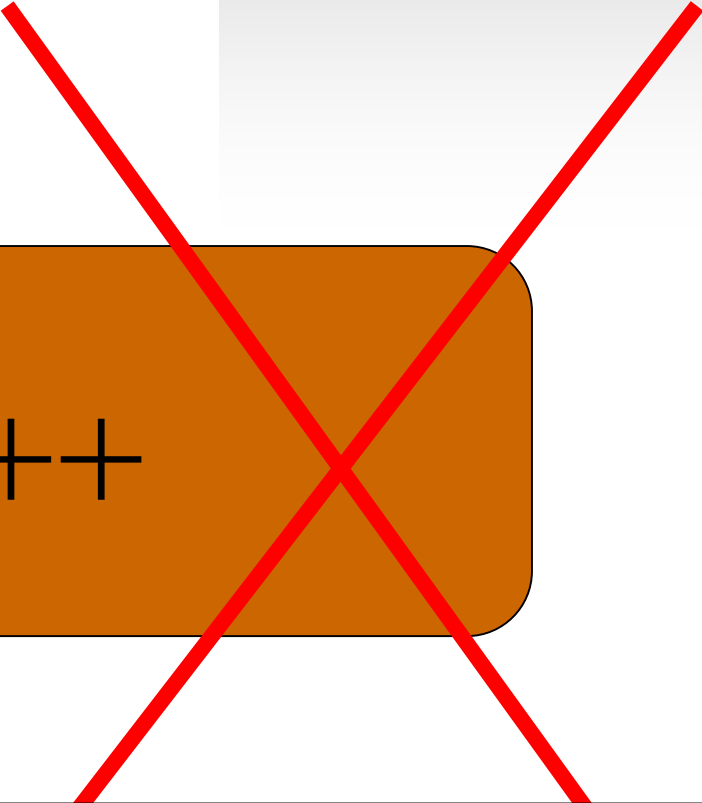
? j,i

A) j=4 i=4

B) j=3 i=4

C) j=-3

++和--结合方向
是“自右至左”



$j = -i++$

最好不要写这样难懂的表达式

(三)、赋值运算符和赋值表达式

赋值运算符

简单赋值运算符：=

复合赋值运算符: +=, -=, *=, /=, %=

- 简单赋值运算符的一般形式为：**变量 = 表达式**
它的作用是将一个表达式的值赋给一个变量。

- 复合赋值运算的一般格式为:

变量 双目运算符= 表达式



复合赋值运算符

它等价于：变量 = 变量 **双目运算符 (表达式)**。只有当表达式简化为一个变量或一个常数时，两边的括号可以省略。

赋值表达式

格式：<变量> <赋值运算符> <表达式>

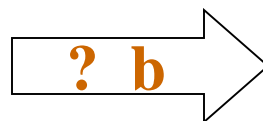
赋值表达式的值：被赋变量的值

类型转换

转换条件：当赋值运算符两侧的数据类型不一致时
转换原则：转换为被赋值变量的类型。

例1

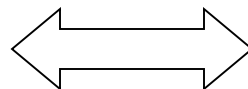
```
float a; int b;  
a=1.2;  
b=a*3;
```



```
b=3
```

例2

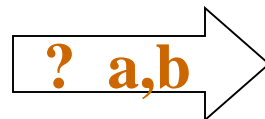
```
a+=3;  
x*=y+8
```



```
a=a+3;  
x=x*(y+8)
```

例3

```
a=b=5;  
a=7+(b=8)
```



```
a=5 ;b=5  
a=15;b=8
```

自右而左
的结合性

(四)、逗号运算符和逗号表达式

逗号运算符

,

逗号表达式

表达式1, 表达式2, , 表达式n

说明

- 1、逗号表达式的求解过程为自左至右，依次计算各表达式的值，最后一个表达式的值即为整个逗号表达式的值；
- 2、逗号运算符的优先级最低。
- 3、使用逗号表达式的目的通常是想分别得到各个表达式的值，而并非一定要得到整个表达式的值。
- 4、常用于for循环语句中，除此以外很少使用。

例1

a=3
a=(3*5, a*4)

? a

a=12

例2

a=3
a=3*5, a*4

? a

a=15
表达式的值为**60**

例3

(a=3*5, a*4), a+5

? a

a=15
表达式的值为**20**

五、不同类型数据间的转换

- 隐式转换（自动转换）
- 显式转换（强制转换）

类型转换——隐式转换（自动转换）

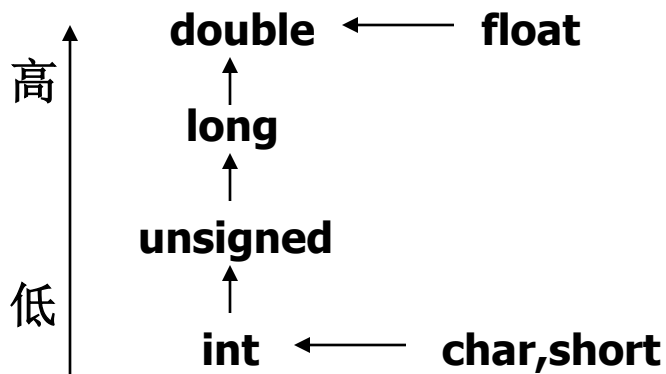
转换发生条件

先转换、后运算

- ❖ **运算转换**-----不同类型数据混合运算时
- ❖ **赋值转换**-----把一个值赋给与其类型不同的变量时
- ❖ **输出转换**-----输出时转换成指定的输出格式
- ❖ **函数调用转换**-----实参与形参类型不一致时转换

类型转换——隐式转换（自动转换）

转换规则



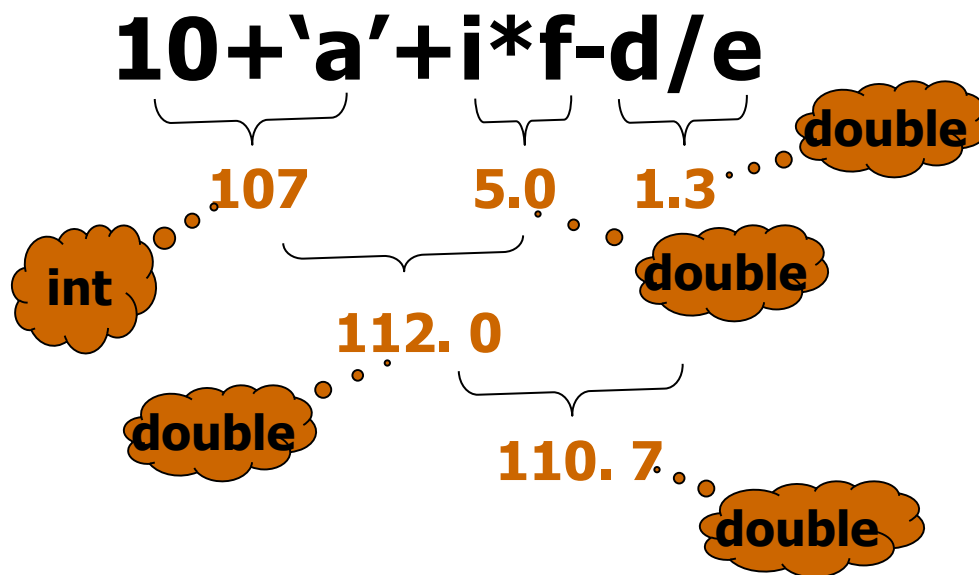
说明:

- ← 必定的转换
- ↑ 运算对象类型不同时转换

- 1、纵向向上的箭头表示不同类型的转换方向（由低类型转化为高类型），不表示转换所经的步骤。
- 2、横向向左的箭头表示必须的转换。

思考： 根据已知变量定义及赋值语句，
计算以下表达式的值，并说明每一步所得
结果的数据类型。

```
int i;  
float f;  
double d;  
long e;  
.....  
i=2;  
f=2.5;  
d=3.9;  
e=3;  
.....
```



类型转换——显式转换（强制转换）

一般形式

(要转换成的数据类型)(被转换的表达式)

注：当被转换的表达式是一个简单表达式时，外面的一对圆括号可以缺省。

举例

```
float x,y;    x=2.3 ; y=4.5;  
(int)(x+y)   //  6  (int)  
(int)x+y     //  6.5 (double)
```

说明

强制转换得到的是所需类型的中间变量，原变量或表达式的类型不变。

思考： 以下程序的输出结果是什么？

```
#include <stdio.h>
void main()
{
    float x;
    int i;
    x=3.6;
    i=(int)x;
    printf("x=%f,i=%d",x,i);
}
```

程序运行结果：

x=3.600000,i=3

结论： 较高类型
向较低类型转换
时可能发生**精度**
损失问题

举例：已知三角形的边长a、b、c，求三角形周长的一半。

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    float a,b,c,s;
```

```
    a=3;b=4;c=5;
```



```
    printf("s=%8.2f",s);
```

```
    return 0;
```

改为： 1.0/2 (1/2.0)
或 (float)1/2 (1/(float)2)

程序运行结果：

s= 0.00

程序运行结果：

s= 6.00

练习

- 已知 $y=3a+2b+c$
- 从键盘输入a、b、c的值
- 输出y 的值

- 输入：
 - 1 2 3
- 输出：
 - 10

顺序结构程序设计总结

- 程序的执行流程是从上之下一步步执行
- 重点掌握
 - 输入和输出函数
 - scanf函数
 - printf函数
 - 数据类型及各种数据类型的输入和输出
 - 运算符
 - 字符型变量的两个特殊的函数
 - getchar()
 - putchar()

(五) 关系运算符和关系表达式 (P91)

关系运算符

< <= > >= == != (6种)

优先级

赋值 < 关系(后2<前4) < 算术

算术、关系、
逻辑、赋值、
字符。

关系表达式

用关系运算符将两个表达式连接起来的式子。

5>3 ?

值

真(1) 假(0)

举例

已知: (a=3 b=2 c=1 d=0)

a==b>c

d=a>b+c?

d==a>b+c

0

1

(六)、逻辑运算符和逻辑表达式

逻辑运算符

! (逻辑非) && (逻辑与) || (逻辑或)

单目

双目

P93,94

运算规则

- (1) !: 取逻辑值的相反值。
- (2) &&: 参加运算的两个逻辑值都为真时, 结果为真(1)。
- (3) ||: 参加运算的两个逻辑值都为假时, 结果为假(0)。

优先次序

赋值 < || < && < 关系 < 算术 < !

P379

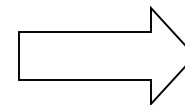
逻辑表达式

用逻辑运算符将关系表达式和逻辑量连接起来的式子。

举例

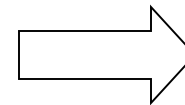
已知: (a= 1 b= 0)

a || b && 0



1

! a && (5>3) || b

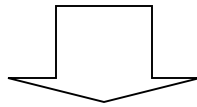


0

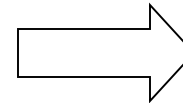
(六)、逻辑运算符和逻辑表达式

举例

5>3 && 8 < 4- !0

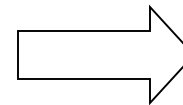


(5>3) && (8 < (4- !0))



0

4 && 5>3 || 2



1

逻辑运算符两侧的运算对象不但可以是 0 和 1，也可以是 0 或非 0 的整数，也可是任何类型的数据。系统判定标准：0 为假；非 0 为真。