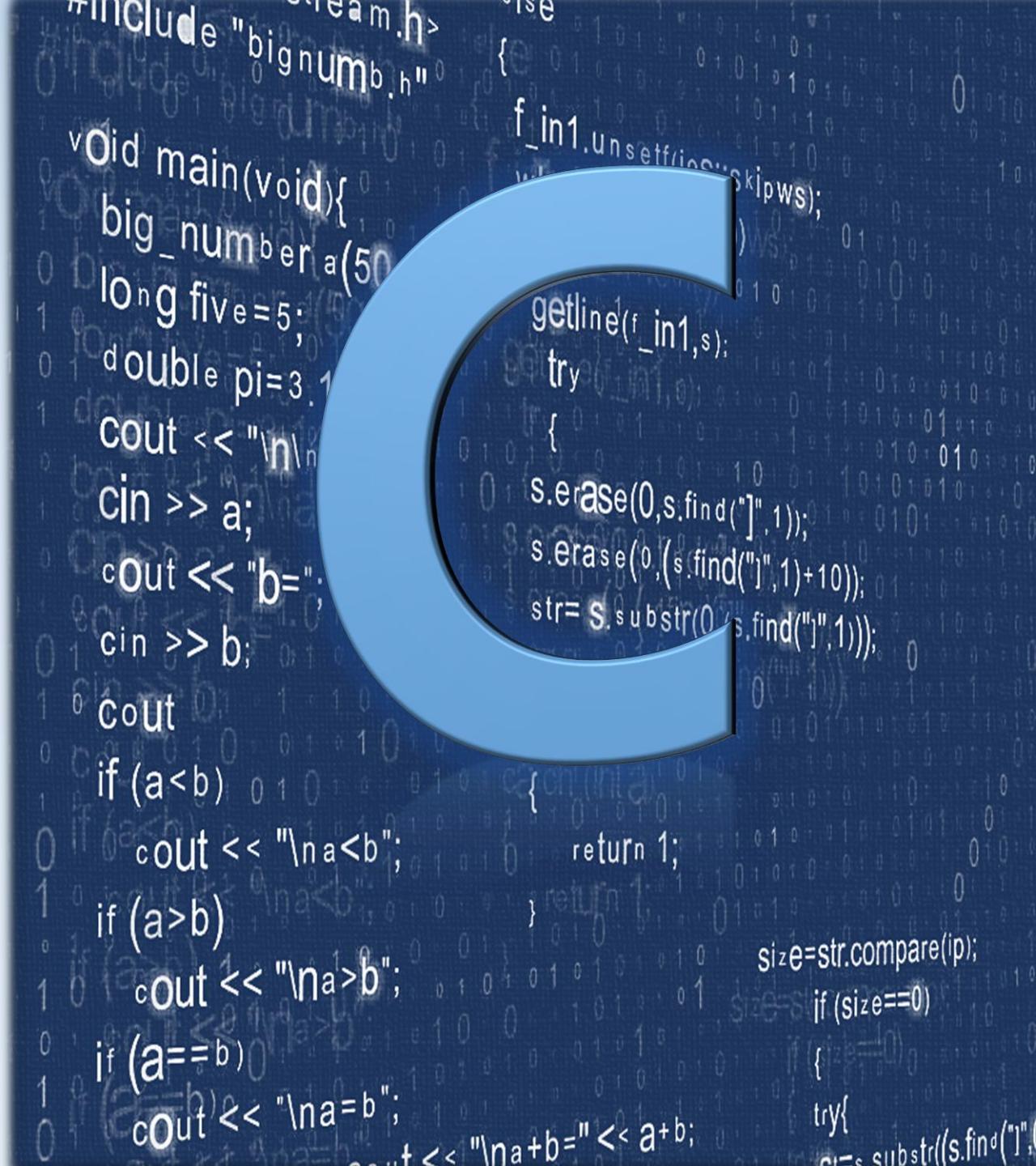


《C语言程序设计》

C语言课程组



上一讲知识复习

- ◆什么是程序
- ◆两种注释方法
- ◆main函数写法
- ◆c程序开发步骤

本讲教学目标

- ◆掌握C语言中的数据类型及区别
- ◆掌握定义变量的方法
- ◆掌握命名规则
- ◆掌握不同类型字面值的写法。
- ◆掌握输入与输出的方式

本讲授课内容

问题求解与算法

数据如何在计算机中表示

数据类型

常量与字面值

数据的输出与输入



问题求解与算法

- ❖任意给定一元二次方程 $ax^2+bx+c=0$ (a 不为零), 设计一个算法, 求解这个方程。

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

问题求解与算法

❖求解上述的算法如下：

step1: 输入a、b、c

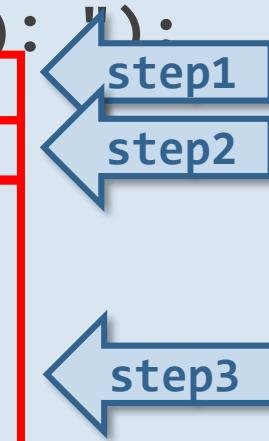
step2: 计算 $\Delta = b^2 - 4ac$

step3: 若 $\Delta \geq 0$, 则 $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

并输出结果，否则无实根。

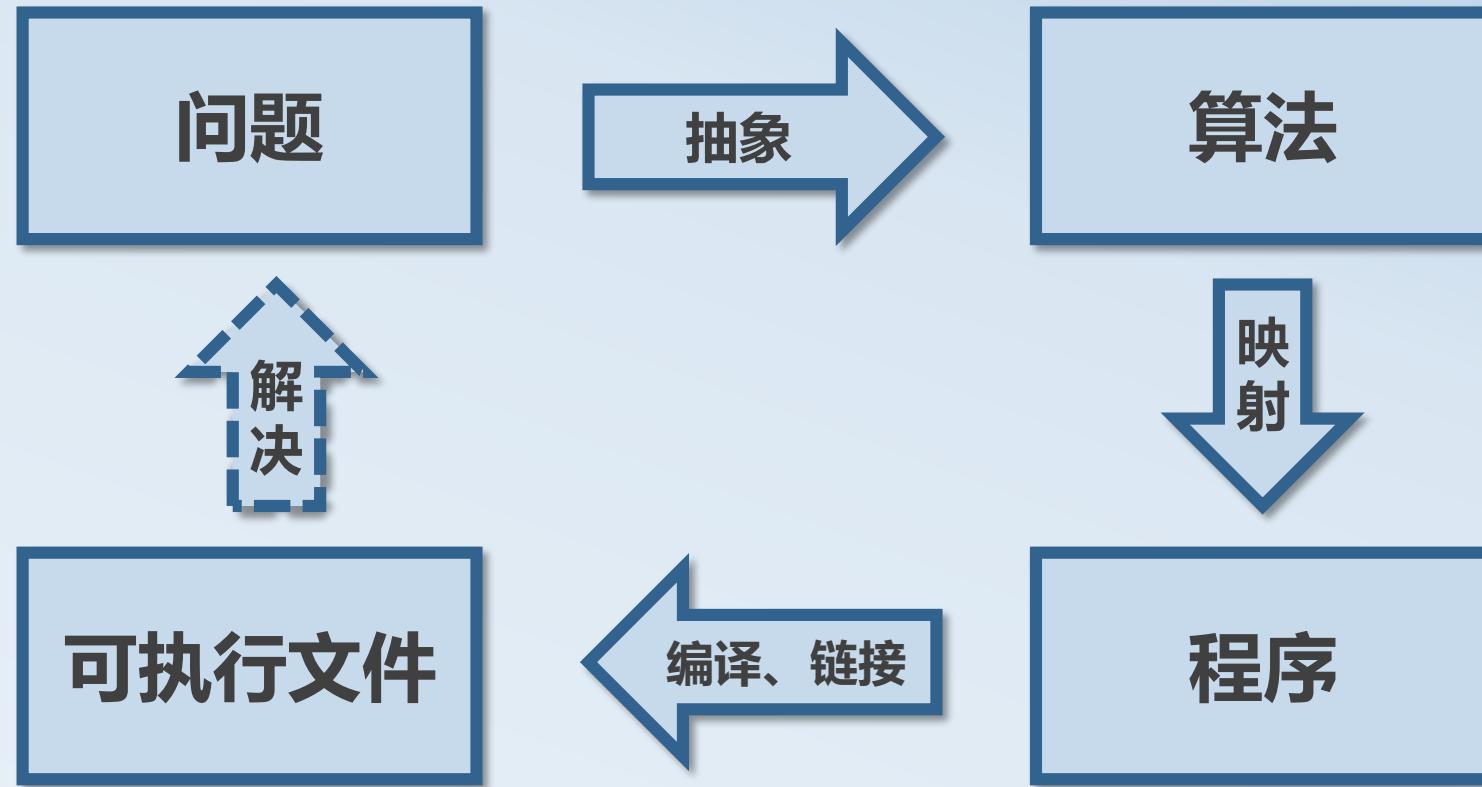
问题求解与算法

```
#include <stdio.h>
#include <math.h> // sqrt函数在此头文件中定义
int main(void)
{
    int a, b, c;
    double delta;
    double x1, x2;
    printf("输入a, b, c(a不为0, 数据间以空格隔开):");
    scanf("%d %d %d", &a, &b, &c);
    delta = b * b - 4 * a * c;
    if(delta >= 0)
    {
        x1 = (-b + sqrt(delta))/(2.0 * a);
        x2 = (-b - sqrt(delta))/(2.0 * a);
    }
    else
    {
        printf("方程无实根。 \n");
    }
    return 0;
}
```



问题求解与算法

- ❖ 从问题到求解的大致过程



问题求解与算法

❖写出求 $1+2+3+\dots+n(n=5)$ 的算法。

◆ 逐一相加法:

step1: $x_1 = 1+2$

step2: $x_2 = x_1+3$

step3: $x_3 = x_2+4$

step4: $x_4 = x_3+5$

◆ 求和公式法:

step1: 取n为5

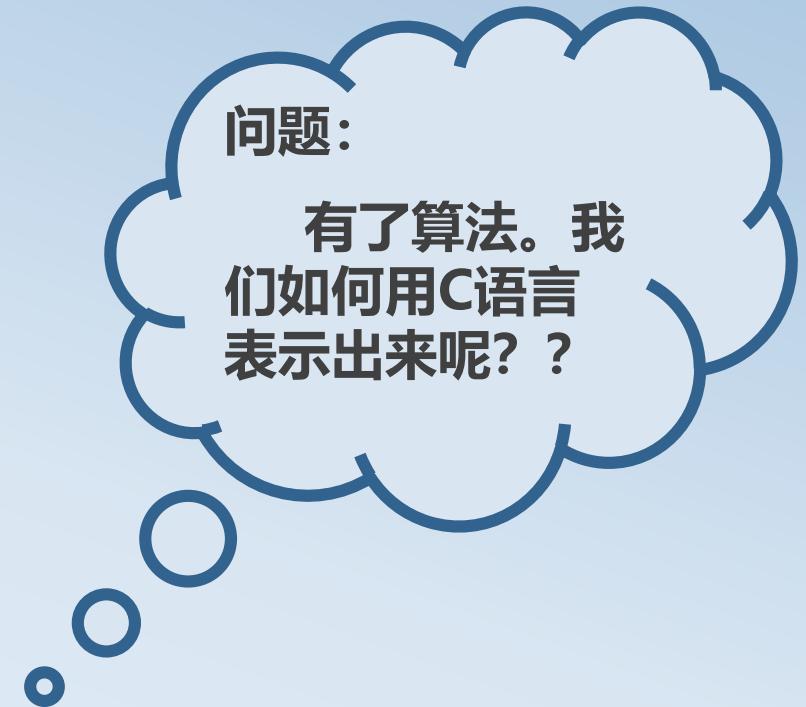
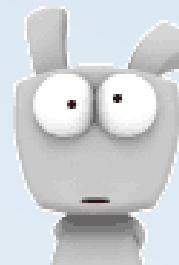
step2: 计算 $sum = n*(n+1)/2$

step3: 输出sum的值

问题求解与算法

❖ 算法的特征：

- ◆ 有穷性
- ◆ 确切性
- ◆ 可行性
- ◆ 一个算法有零个或多个输入
- ◆ 一个算法有一个或多个输出



本讲授课内容

问题求解与算法

数据如何在计算机中表示

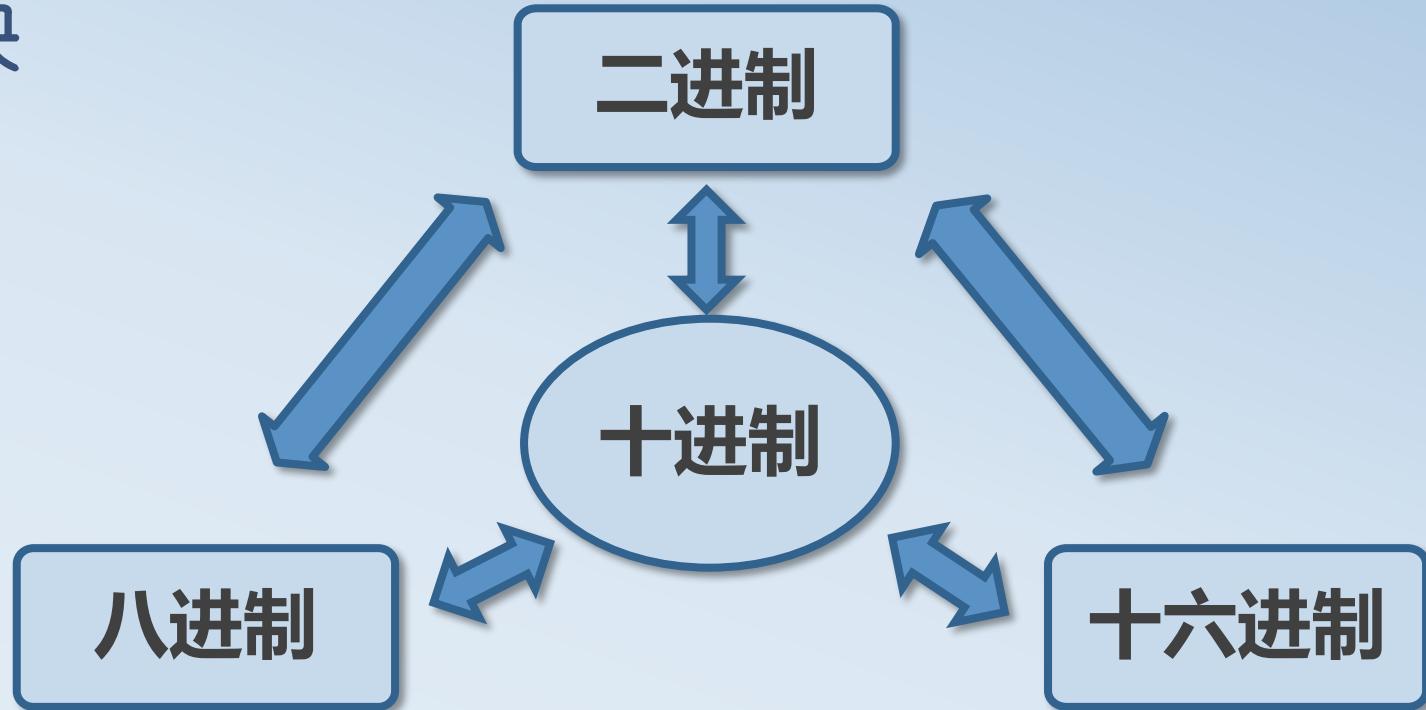


数据类型

常量与字面值

数据的输出与输入

进制的转换



二进制:	0	1														
八进制:	0	1	2	3	4	5	6	7								
十进制:	0	1	2	3	4	5	6	7	8	9						
十六进制:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f

内存结构



◆ 常用的表示存储空间大小单位：

b, B, KB, MB, GB, TB

◆ 一个位有多大？

- 只能是“0”或者“1”，这叫二进制

◆ 一个字节有多大？

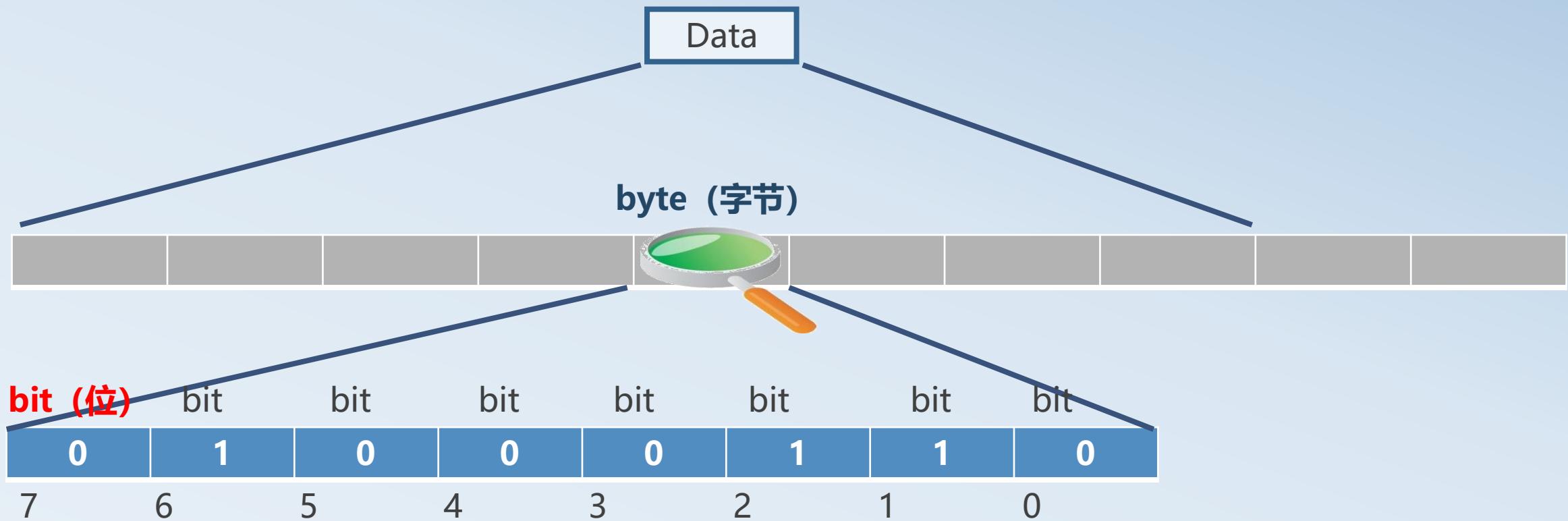
- 8位二进制数
- 保存一个字符（英文字母、数字、符号）
- 可以表示0~255之间的整数

◆ 内存以字节编址

0012FF78
0012FF79
0012FF7A
0012FF7B
0012FF7C
0012FF7D
0012FF7E
0012FF7F
0012FF80
0012FF81

⋮

内存结构



数据如何在计算机中表示

❖求两个数的和、差、积、商。

- 计算机如何表示数？
- 计算机能够表示年龄吗？
- 计算机能够表示工资吗？
- 计算机能够表示字母吗？
- 计算机能够表示人名吗？
-

❖程序和数据

- 程序中常量、变量表示数据。
- 整型数据 int
- 实数 float
- 字符 char
- 字符串 char[20]
- 用户可以自定义自己的类型。

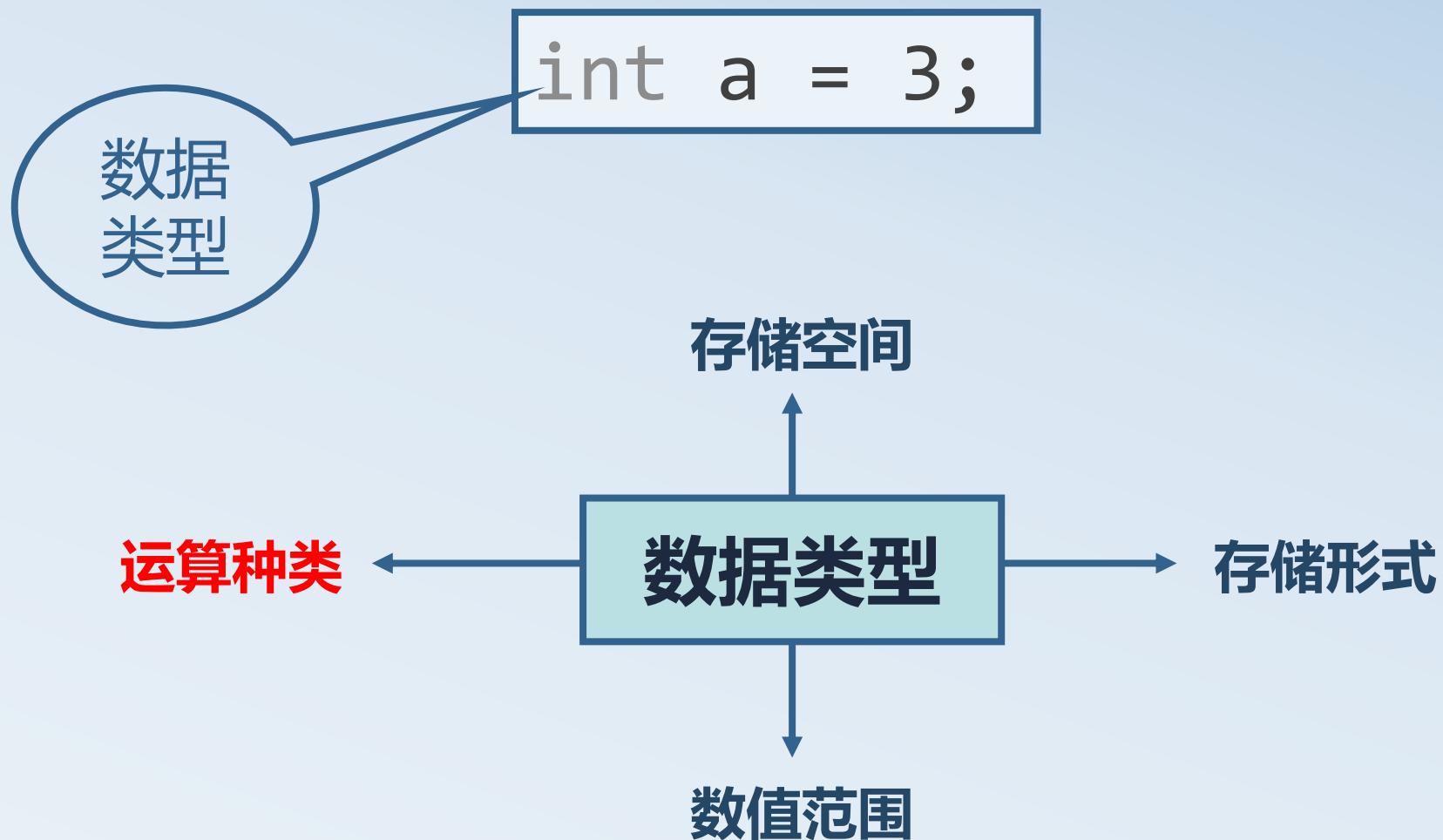
❖程序中通过数据类型定义变量来表示自己需要的数据

❖可以给内存中**一块连续的空间**起一个名字，通过这个名字来操作它，这个名字就是变量名，也叫做标识符。

数据如何在计算机中表示

```
/*计算两个数的和并输出*/  
#include <stdio.h>  
int main(void)  
{  
    int add1, add2, sum;  
  
    add1 = 21;  
    add2 = 34;  
    sum = add1 + add2; // 计算add1+add2的和赋给sum  
    printf("The sum is %d", sum);  
    return 0;  
}
```

数据如何在计算机中表示



使用sizeof测试数据类型长度

- **sizeof**可以得到变量或数据类型占用的字节数
- 基本形式：
 - sizeof(数据值)
 - sizeof(数据类型)
- 举例：

```
int i = 2;  
  
sizeof(int);           // 结果4  
sizeof(i);            // 结果4  
sizeof(2);            // 结果4
```

数据如何在计算机中表示

❖ 变量声明的基本语法：

数据类型 变量名1, 变量名2 ..., 变量名n;

unsigned int age = 3;

数据类型

变量列表

extern const unsigned long int x, y;

(带附加声明) 数据类型

变量列表

数据如何在计算机中表示

- ❖ 类型说明：

- ◆ **类型说明符**
- ◆ 存储类型说明符
- ◆ 类型限定说明符



- ❖ 变量名 (标识符、变量名)

- ◆ 变量名由数字，字母和下划线组成
- ◆ 数字不能打头
- ◆ 不能使用C语言中的关键字
- ◆ C语言的变量名中的字母区分大小写
- ◆ 变量名应该尽量有意义，参考编程规范

数据如何在计算机中表示

❖ 标识符辨认

符合规则的命名

GoodName

goodname

_ma_ma_

Cup123

F1f4_333

非法的名称

int

123f

@edu2act.org

\$php6

!printf

数据如何在计算机中表示

- ❖ 未初始化的变量的值是随机数

- ❖ 在定义时初始化赋值

- 例: int num = 10; /*好的风格*/
int num1 = 1, num2, num3; /*坏的风格*/
int num1 = num2 = 2; /*错误的赋值*/

- ❖ 在定义后赋值

- 例: int num;
num = 10;

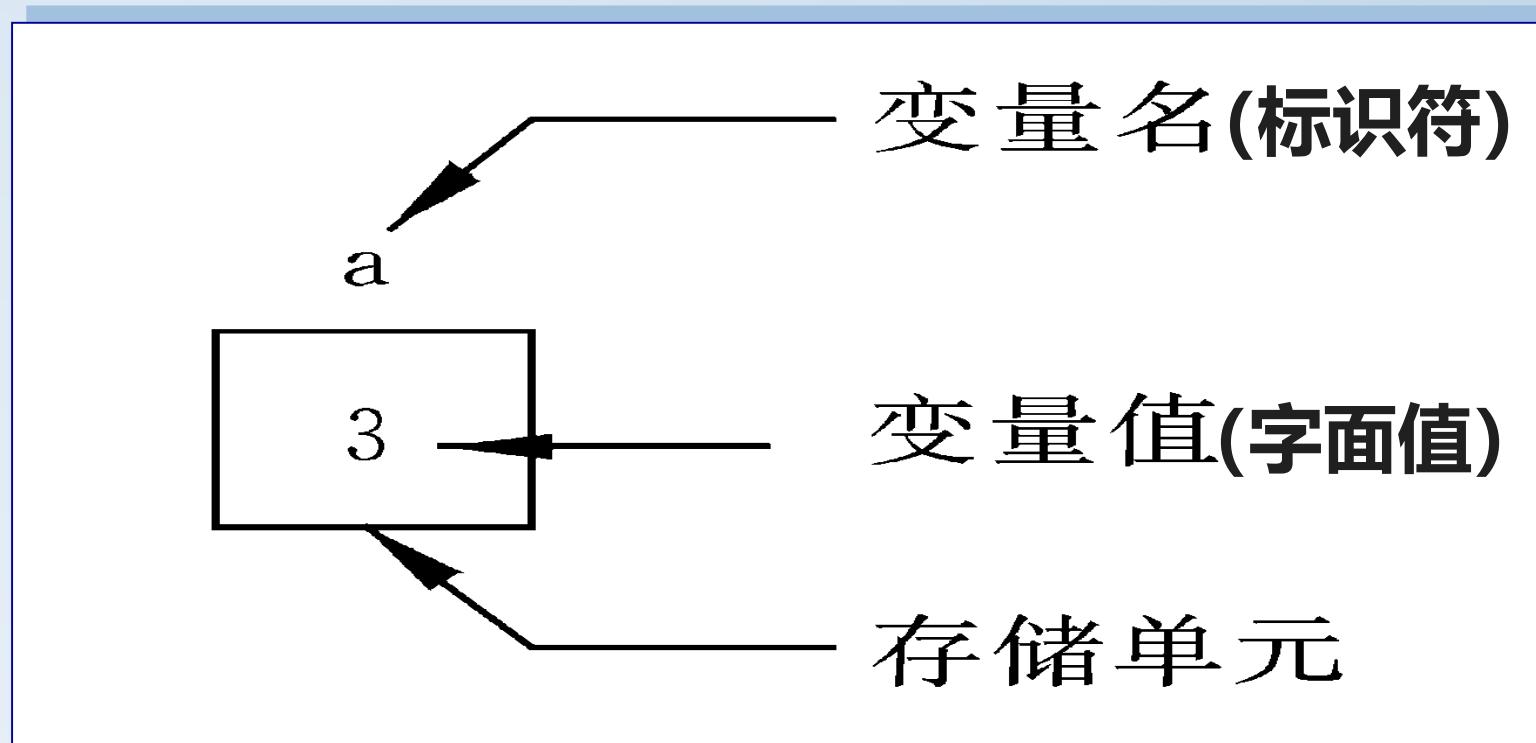
- ❖ 变量的使用规则

- 先定义, 后使用
 - 先赋值, 后参加运算

数据如何在计算机中表示

```
int a = 3;
```

0x0012FF1C
变量地址



数据如何在计算机中表示

```
#include <stdio.h>

int main(void)
{
    int student, age;
    int if = 1;
    float score = 90.5;

    student = 2;
    Age = 20;

    student = Age + 10;
    printf("%d %d %d %f", if, student, age, score);

    return 0;
}
```

数据如何在计算机中表示

❖ 变量声明小结

- 1. 变量名符合命名规范。
- 2. 可以在定义变量的同时进行初始化。

例如： int x = 5;

- 3. 对于变量一定要 “**先定义,后使用**” 。
- 4. 变量一定要 “**先赋值后参加运算**” 。
- 5. 同一个命名空间内变量**不能重名**。

本讲授课内容

问题求解与算法

数据如何在计算机中表示

数据类型

常量与字面值

数据的输出与输入

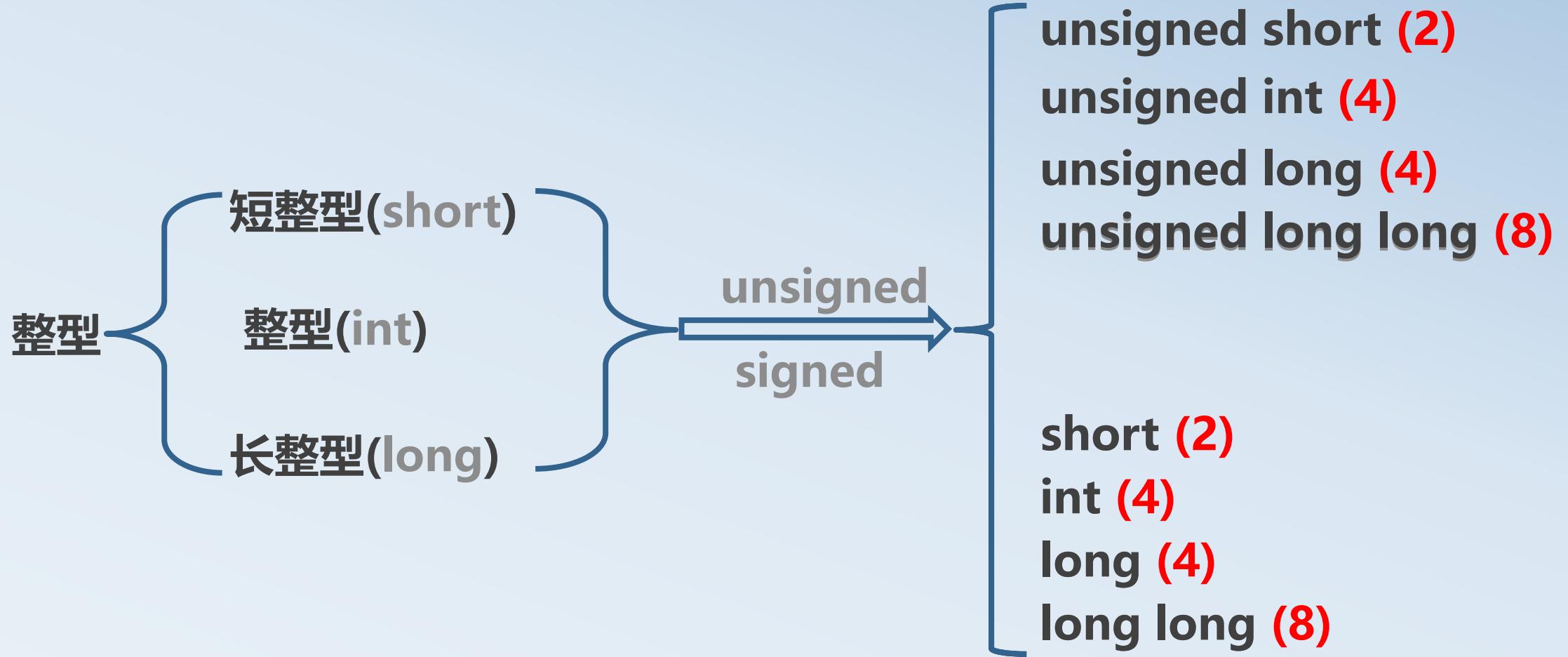


数据类型

- 数据类型介绍



数据类型

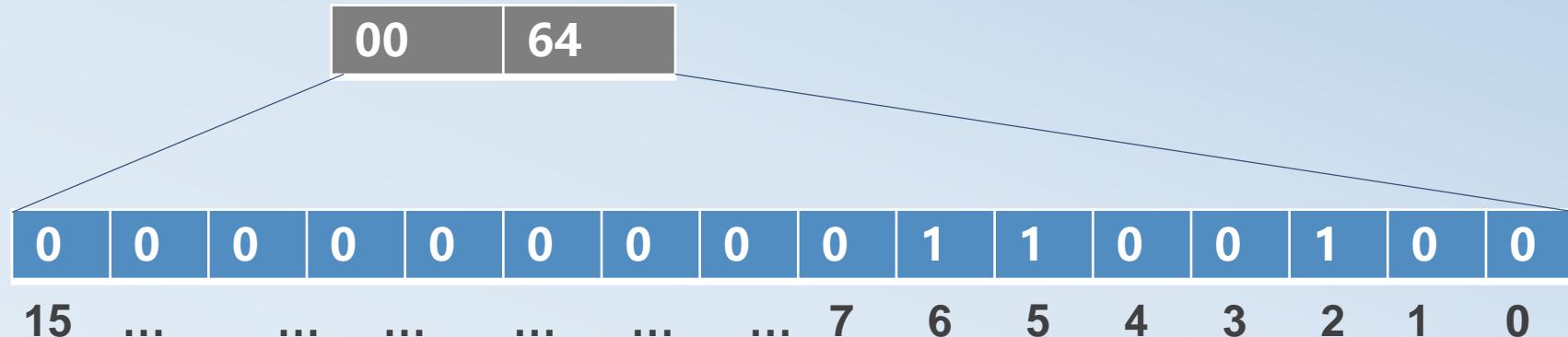


整数在内存中表示形式

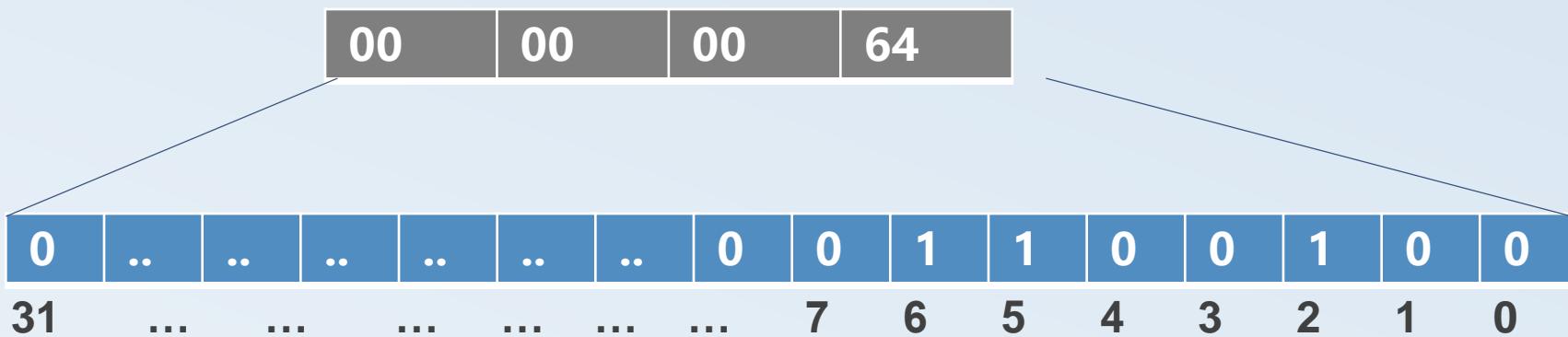
- ❖ 无符号整数在内存中表示
- ❖ 有符号整数在内存中表示

无符号整数在内存中的表示

❖ 无符号整数的表示 `unsigned short a = 100;`

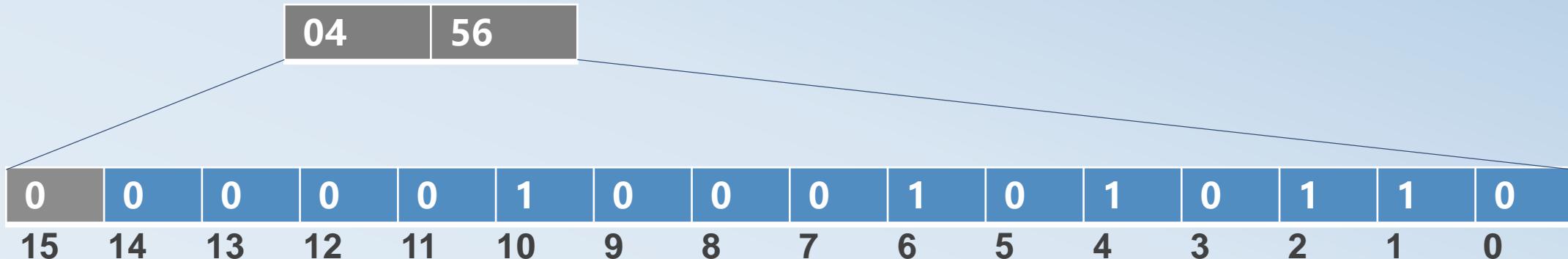


❖ 无符号整数的表示 `unsigned int a = 100;`

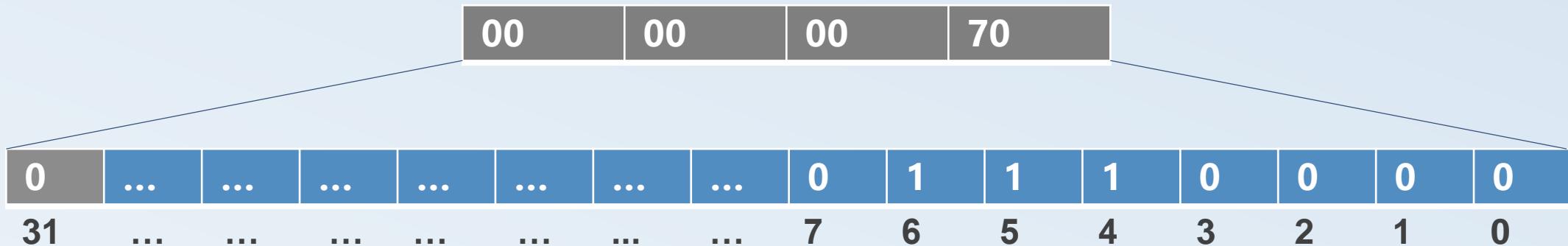


有符号整数在内存中的表示

❖ 短整型(signed short)



❖ 整型、长整型(signed int、 signed long)



整数在内存中的表示



- ❖ 在内存中数值是以**补码**的形式存储的
- ❖ 有符号整数在内存中的存储
 - 正数：
 - 原码、补码和反码相同
 - 负数：
 - 原码：数值的二进制表示
 - 反码：符号位不变，数值的二进制按位取反
 - 补码：数值的反码加1

数据在内存中的表示

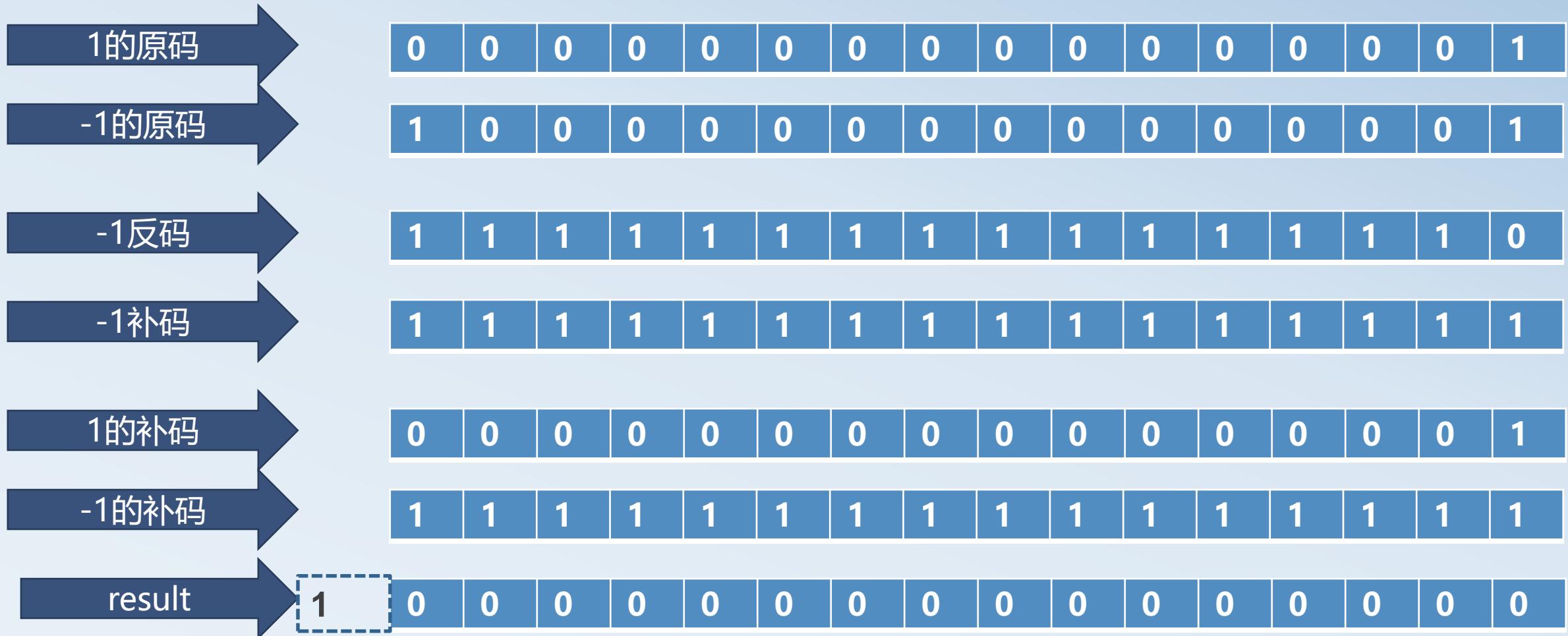
❖ 要求

计算1和-1的和

❖ 目的

揭示正数和负数的内存表示

分析



数据在内存中的表示

```
#include <stdio.h>

int main(void)
{
    short num1 = 1;
    short num2 = -1;
    short result;
    result = num1 + num2;
    printf("%d\n", result);
    return 0;
}
```

整型小结

类型	符号	关键字	所占位数	数的表示范围
整型	有	(signed)short	16	$-2^{15} \sim 2^{15} - 1$
		(signed)int	32	$-2^{31} \sim 2^{31} - 1$
		(signed)long	32	$-2^{31} \sim 2^{31} - 1$
		(signed)long long	64	$-2^{63} \sim 2^{63} - 1$
	无	unsigned short	16	$0 \sim 2^{16} - 1$
		unsigned int	32	$0 \sim 2^{32} - 1$
		unsigned long	32	$0 \sim 2^{32} - 1$
		unsigned long long	64	$0 \sim 2^{64} - 1$

$\text{sizeof(short)} \leq \text{sizeof(int)} \leq \text{sizeof(long)} \leq \text{sizeof(long long)}$

数据类型

❖ 数据的范围--<limits.h>

```
/* limits.h中的部分内容 */  
  
#define SHRT_MIN           (-32768)  
#define SHRT_MAX            32767  
#define USHRT_MAX           0xffff  
#define INT_MIN              (-2147483647L-1)  
#define INT_MAX              2147483647  
#define UINT_MAX             0xffffffff  
#define LONG_MIN             (-2147483647L-1)  
#define LONG_MAX             2147483647L  
.....
```

整型溢出

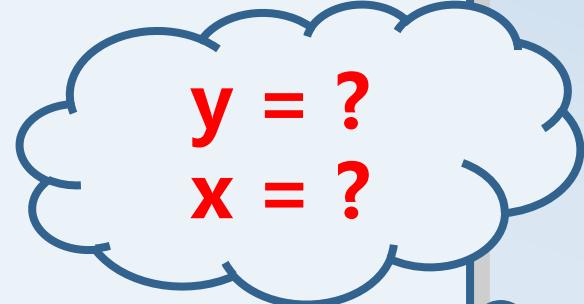
```
#include <stdio.h>

int main(void)
{
    short y = -32768;
    unsigned short x = 65535;

    y = y - 1;
    printf("%d\n", y);

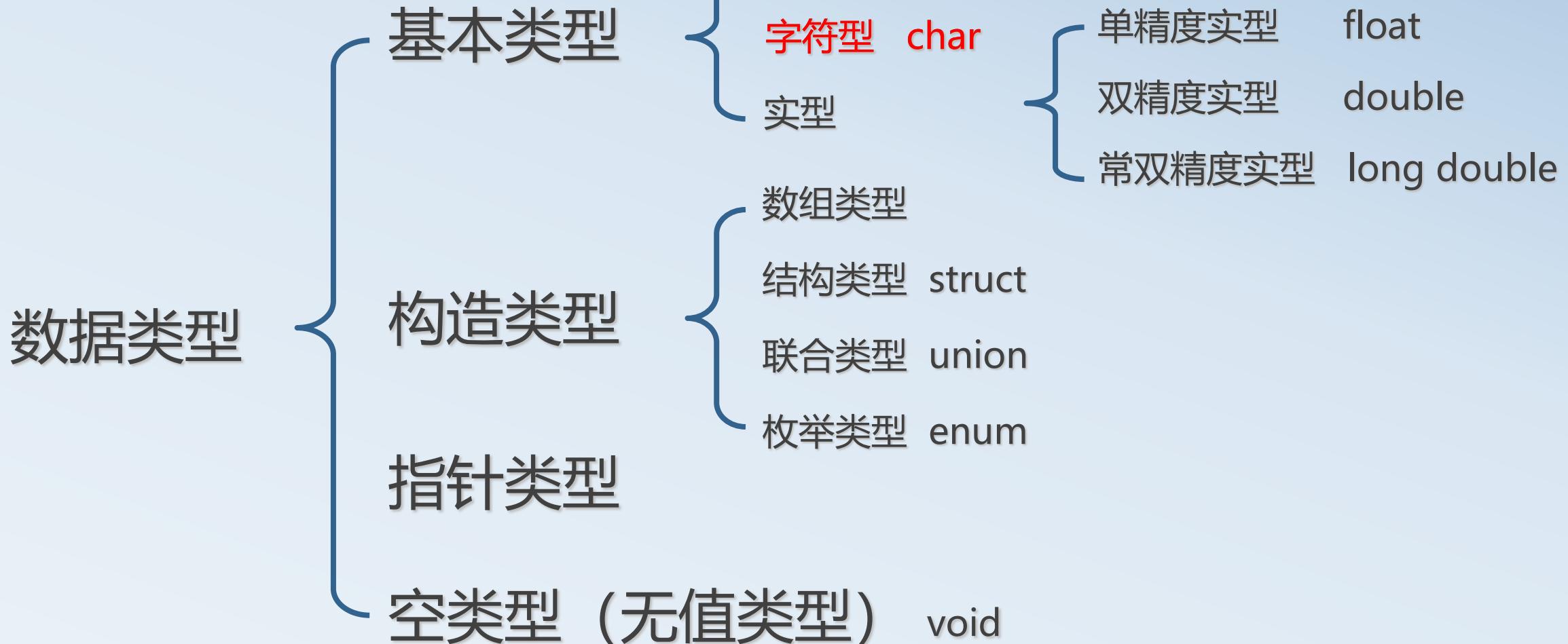
    x = x + 1;
    printf("%u\n", x);

    return 0;
}
```



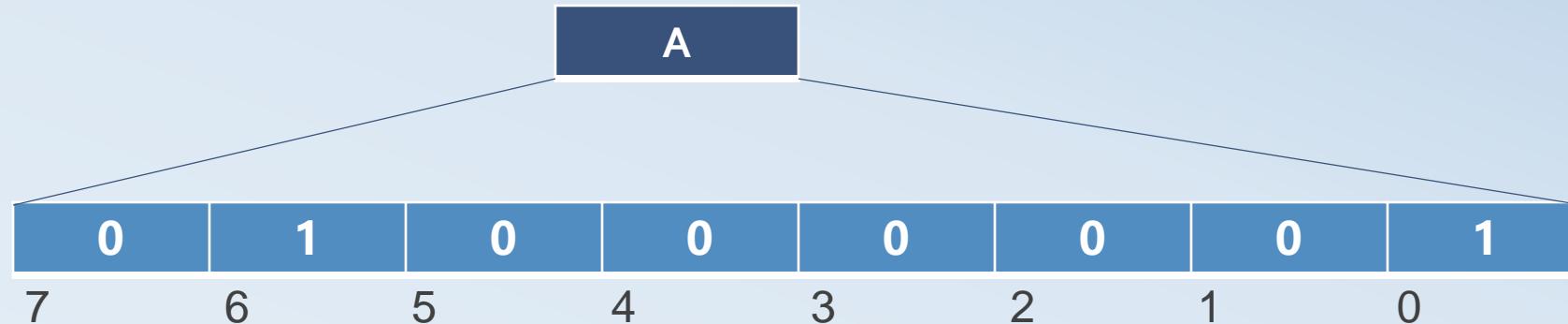
数据类型

- 数据类型介绍



char类型

- ❖ char型的数据在内存中保存的是字母的ASCII码
 - vs2012中字符型是按补码存放的(signed char)



- ❖ 补充：ASCII表(美国信息交换标准代码)American Standard Code for Information Interchange

ASCII表

ASCII码	控制符号	ASCII码	字符符号	ASCII码	字符符号	ASCII码	字符符号
0	NUL (空白)	32	空格	64	@	96	`
1	SOH (序始)	33	!	65	A	97	a
2	STX (文始)	34	"	66	B	98	b
3	ETX (文终)	35	#	67	C	99	c
4	EOT (送终)	36	\$	68	D	100	d
5	ENQ (询问)	37	%	69	E	101	e
6	ACK (应答)	38	&	70	F	102	f
7	BEL (告警)	39	'	71	G	103	g
8	BS (退格)	40	(`)	72	H	104	h
9	HT (换表)	41	\t	73	I	105	i
10	LF (换行)	42	\n	74	J	106	j
11	VT (纵表)	43	\r	75	K	107	k
12	FF (换页)	44	\f	76	L	108	l
13	CR (回车)	45	\v	77	M	109	m
14	SO (移出)	46	\u0000	78	N	110	n
15	SI (移入)	47	/	79	O	111	o
16	DLE (转义)	48	0	80	P	112	p
17	DC1 (设控 1)	49	1	81	Q	113	q
18	DC2 (设控 2)	50	2	82	R	114	r
19	DC3 (设控 3)	51	3	83	S	115	s
20	DC4 (设控 4)	52	4	84	T	116	t
21	NAK (否认)	53	5	85	U	117	u
22	SYN (同步)	54	6	86	V	118	v
23	ETB (组织)	55	7	87	W	119	w
24	CAN (作废)	56	8	88	X	120	x
25	EM (单尾)	57	9	89	Y	121	y
26	SUB (取代)	58	=	90	Z	122	z
27	ESC (换码)	59	?=	91	[123	{
28	FS (卷隙)	60	<=	92	\	124	
29	GS (勘隙)	61	=	93]	125	}
30	RS (录隙)	62	>=	94	^	126	~
31	US (元隙)	63	?	95	_	127	DEL (删除)

char类型内存占位

```
char a1 = 'A';    ↔ char a1 = 65;  
  
printf(" a1 = %c ", a1);  
printf(" a1 = %d ", a1);
```

注：有些以“\”开头的特殊字符称为转义字符

'\n'	换行
'\t'	横向跳格
'\r'	回车
'\\'	反斜杠

字符型与整型的关系

- 以字符型和整数型两种格式输出字符变量

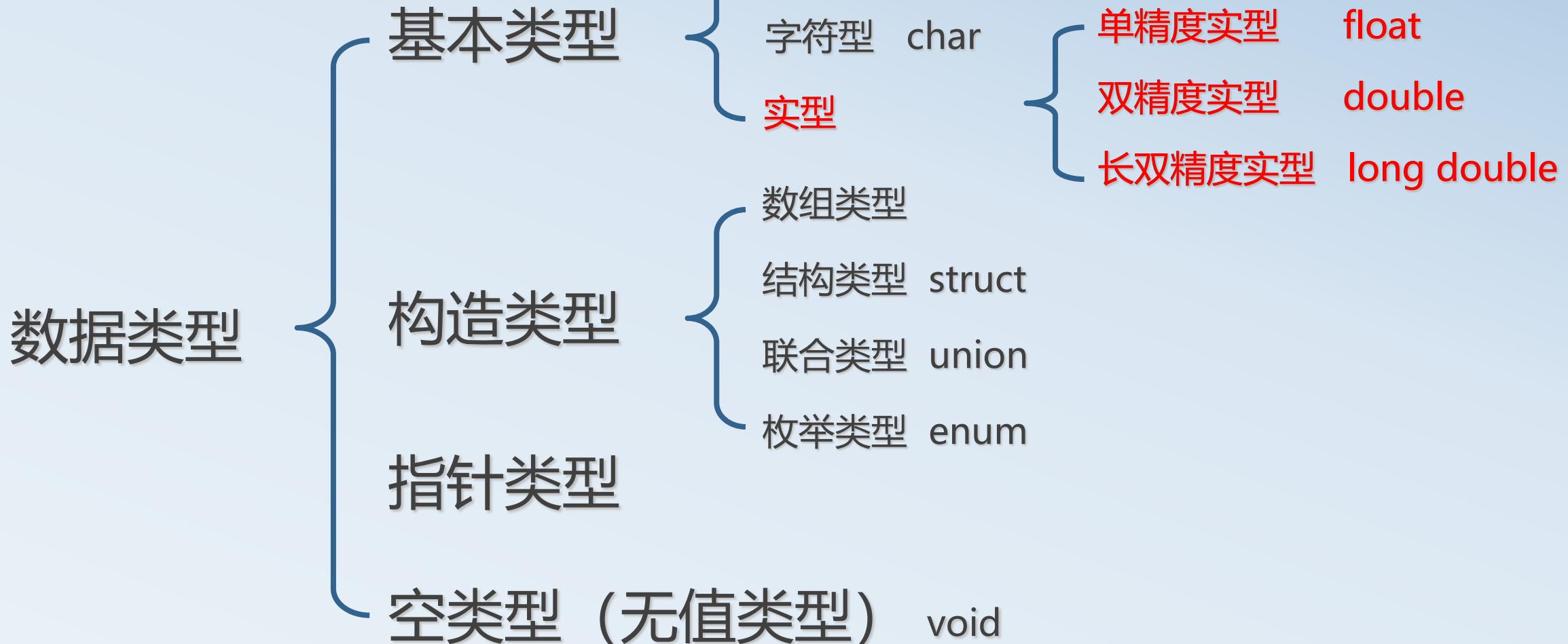
```
#include <stdio.h>
int main(void)
{
    char ch = 'a';                  /*定义 ch 为字符型变量*/
    printf("%c, %d\n", ch, ch);/*以字符、整数形式输出ch */
    return 0;
}
```

- 小写字母转换为大写字母

```
#include <stdio.h>
int main(void)
{
    char ch = 'b';
    ch = 'b' - 32;
    printf("%c, %d\n", ch, ch);
    return 0;
}
```

数据类型

- 数据类型介绍



实型内存占位

◆ 实型

$$22.625 = (10110.101)_b = (1.0110101)_b \times 2^4$$

$$N = S \times r^j$$

符号	阶码j	尾数S
	取值范围	有效位数
22.625	0 10000011	011010100000000000000000

实数范围

类型	符号位	关键字	位数	范围
实数	有	float	32	-10 ³⁸ ~10 ³⁸
	有	double	64	-10 ³⁰⁸ ~10 ³⁰⁸
	有	long double	64	-10 ³⁰⁸ ~10 ³⁰⁸

`sizeof(float) ≤ sizeof(double) ≤ sizeof(long double)`

实型范围

◆ 数据的范围 <float.h>

/* float.h 中的部分内容 */	
#define FLT_DIG	6
#define FLT_EPSILON	1.192092896e-07F
#define FLT_MANT_DIG	24
#define FLT_MAX	3.402823466e+38F
#define FLT_MAX_10_EXP	38
#define FLT_MAX_EXP	128
#define FLT_MIN	1.175494351e-38F
#define FLT_MIN_10_EXP	(-37)
#define FLT_MIN_EXP	(-125)
.....	

实型数据小结

- ❖ 实型数据无法精确表示所有的数字。
- ❖ 每一种实型数据都有自己的有效位数和精度。
- ❖ 实型数据**无法直接判等**。

数据类型

❖ 根据实际需要设计相应类型的变量

- ◆ char style = 'A' ;
- ◆ short age = 3;
- ◆ int income = 2000;
- ◆ long stars = 3799900;
- ◆ float money = 6.32;
- ◆ double distance;
- ◆ long double root;
- ◆

数据类型练习

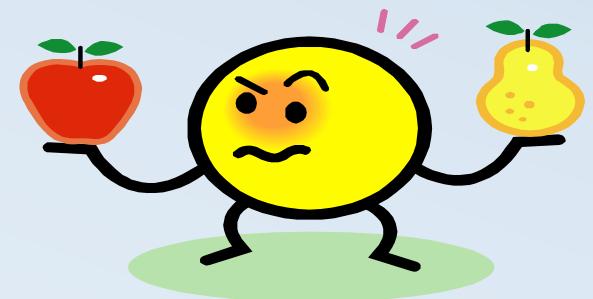
例： 编程输出 $1+2+\dots+n$ 的值，n 由用户从键盘上输入。

```
#include <stdio.h>
int main(void)
{
    unsigned int n;
    unsigned long long sum;
    printf("请输入n值:");
    scanf("%u", &n);
    sum = n * (n + 1) / 2;
    printf("1+2+...+n(n=%u)=%llu.\n", n, sum);

    return 0;
}
```

数据类型

- ❖ 每种数据类型所占内存字节数、所能表示的数据范围、数据表示的形式以及这个类型数据所能进行的操作都是不同。
- ❖ 要根据实际需要选择合适的数据类型定义变量。



本讲授课内容

问题求解与算法

数据如何在计算机中表示

数据类型

常量与字面值

数据的输出与输入



常量与字面值

```
#include <stdio.h>

int main(void)
{
    const double PI = 3.14;
    double area = 0.0;
    double r = 10.0;

    area = PI * r * r;
    PI = 3.14159; // error

    return 0;
}
```

常量与字面值

❖ 当变量声明中包括类型限定符 “**const**” 关键字时，可以声明**常量**。

❖ 字面值：

- 整型： 100 0x1EA0 0376
- 浮点型： 1.0 3e2 .3415
- 字符型： 'a' '\'' '\"' '\n'
- 字符串型：“hello” “a” “0” “”

❖ 例如

- `const int x = 1;`
- `const double y = 2;`
- `const float a = 1.0, b = 2.0, c = 3.0;`
- `const unsigned int m = 3, n = 4;`

常量分类

❖ 常量可分为：

- 字面常量（直接常量）
 - 整型常量
 - 实型常量
 - 字符型常量
 - 字符串常量
- 符号常量
- const常量

常量与字面值

❖ 字面值类型:

- ◆ 整型字面值
- ◆ 浮点型字面值
- ◆ 字符型字面值
- ◆ 字符串型字面值

字面值类型	字面值	说明
整型字面值	100	
	0x13B4	以 0x 或 0X 开头的整型字面值是十六进制表示
	0347	以数字 0 开头的整型字面值是八进制表示
浮点型	1.0	
	3e2	表示 3×10^2 , 即 300
	1.0E-3	表示 1.0×10^{-3} , 即 0.001
	.3415926	表示 0.3415926
字符型	'a'	
	'b'	
	'c'	
字符串型	"hello world"	
	"请输入 x、y 的值"	

整型字面值

❖ 整型字面值表示:

- 十进制整数: 123, -456, 0
- 八进制整数: 0123, 011
- 十六进制整数: 0x123, 0Xff

❖ 整型常量的类型

- 一般默认整型常量为int型。也可以根据其值所在范围或后缀确定其数据类型
 - 长整型字面值 123L, 23l,
 - 无符号整型字面值 123U, 123u
 - 无符号长整型字面值 123UL, 123ul
 - long long型字面值 123ll, 123LL

浮点型字面值

◆ 浮点型字面值表示:

- 科学计数法: 5e3 , 12.3e3 ,123E2, 1.23e4
- 自然数计数: 5.3 ,0.123, .123, 123.0, 123.
- 默认为**double**型, 也可以使用后缀f, F, l, L确定类型。

字面值	计数法	进制	字面值类型	备注
double x = 30.23;	自然计数法	十进制	double	未使用后缀
double x = 0.3023e2;	科学计数法	十进制	double	未使用后缀
float x = 30.23F;	自然计数法	十进制	float	
long double x = 30.23L;	自然计数法	十进制	long double	
long double x = .3023e2L;	科学计数法	十进制	long double	
long double x = 0X.3023p2L;	科学计数法	十六进制	long double	
double x = 0X.3023p2;	科学计数法	十六进制	double	
float x = 0X.3023p2F;	科学计数法	十六进制	float	

字符型字面值

❖字符型字面值是用单引号括起来单个普通字符或转义字符。

- 执行字符集： 'a' 、 'A' 、 '0' 、 '9' 、 '\$' 等
- 宽字符集 (wchar_t)
- 转义字符：用来表示很难输入的字符

转义字符	含义	转义字符	含义
\n	换行	\t	水平制表
\v	垂直制表	\b	退格
\r	回车	\f	换页
\a	响铃	\\"	反斜线
\ '	单引号	\ "	双引号
\ddd	3位8进制数代表字符	\xhh	2位16进制数代表的字符

常量与字面值

- ❖ 字符串字面值是由一对双引号括起来的字符序列（以' \0' 结束）。
 - “” 表示空串(只包含' \0')。
 - 'a' 和" a" 的不同

字符串字面值	实际存储的字符串	备注
"a"	"a\0"	与'a'不同， "a"中有两个字符：'a'及'\0'
"a \x61 b"	"a a b\0"	
"\141 bc"	"a bc\0"	
"\141""bc\0"	"abc\0"	
"\1414abc"	"a4abc\0"	数字转义字符\141 是八进制表示'a'
"4\xabab"		数字转义字符\xab 是十六进制表示的字面值

注意：

- 1.字符串字面值会自动在字符串最后加上 '\0'，占据一个字节的内存
- 2.求字符串长度的时候注意转义字符占据一个字节，例如：' \xdd' 等

本讲授课内容

问题求解与算法

数据如何在计算机中表示

数据类型

常量与字面值

数据的输出与输入



数据的输出与输入

❖ 所谓输入输出是相对于计算机主机而言的

- **输出**:从计算机向外部输出设备(显示器,打印机)输出数据。
- **输入**:从输入设备(键盘,鼠标,扫描仪)向计算机输入数据。



C语言的输出与输入

- ❖ C语言本身不提供输入输出语句, 输入和输出操作是由C函数库中的函数来实现的
- ❖ 例如:
 - 格式输入函数:scanf 格式输出函数:printf
 - 字符输入函数:getchar 字符输出函数:putchar
 - 字符串输入函数:gets 字符串输出函数:puts
- ❖ 使用这些函数需要包含 “stdio.h” 头文件
 - **printf ↔ scanf**
 - **putchar ↔ getchar**
 - **puts ↔ gets**

printf格式输出

❖ 格式输出函数

- 函数作用：将结果按要求输出到标准设备上（通常是屏幕）。
- 一般形式：
 - printf(字符串字面值, 参数1, 参数2, ..., 参数n);
 - 参数可能为变量、常量、字面值
 - 字符串字面值里的非转义字符和非转换说明符原样输出
 - 返回值 大于0表示输出的字符个数，小于0表示输出错误。
- printf函数支持0到n个参数的输出：
 - printf("hello"); // 0个参数
 - printf("%d" , x); // 1个变量
 - printf("%d" , 8); // 1个字面值
 - printf("x=%d,y=%f" , x, y); // 2个参数

格式转换说明符

printf格式输出

❖ 对printf而言，转换说明的一般形式：

- %[标志字符][最小宽度说明][精度说明][长度修正说明符]<转换操作符>
- 转换说明以**%**开始，依次出现下列元素：
 - 0个或多个**标志字符**（可选）。包括：-、+、0、#或空格。
 - **最小宽度说明**（可选）。用十进制整型字面值或星号表示。
 - **精度说明**（可选）。小数点后加一个十进制整型字面值表示。
 - **长度修正说明符**（可选）。包括：l、L、h、hh、j、x、t。
 - **转换操作符（必选）**。包括：a、A、c、d、e、E、f、g、G、i、n、o、p、s、u、x、X、%。

常用的转换操作符

数据类型	转换操作符	含义
signed int	d、i	对有符号整数进行格式转换
unsigned int	u	对无符号整数进行格式转换
	o	对无符号整数按八进制输出
	x、X	对无符号整数按十六进制输出
float/double	f、F	对浮点数按十进制计数法输出
	e、E	对浮点数按科学计数法输出
	g、G	对浮点数按十进制计数法或科学计数法输出
	a、A	浮点数、十六进制数字和p-记数法 (c99)
char	c	输出一个字符
字符串	s	输出一个字符串
其它	%	输出一个百分号
	n	将输出流里当前的字符个数输出到一个整数里，要求操作数为 有符号数的地址 (了解)

长度修正说明符

◆长度修正说明符有h、l、ll、L等。

数据类型	转换操作符	含义
signed short	hd、hi	对有符号short进行格式转换
signed long	ld、li	对有符号long进行格式转换
signed long long	lld、lli	对有符号long long进行格式转换
unsigned short	hu、ho、hx、hX	对无符号short进行格式转换
unsigned long	lu、lo、lx、lX	对无符号long进行格式转换
unsigned long long	llu、lllo llx、lX	对无符号long long进行格式转换
long double	Lf、LF、Le、LE Lg、LG、La、LA	对long double输出

关于转换操作符的使用

❖ 不匹配的转换操作符，后果不可预料

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

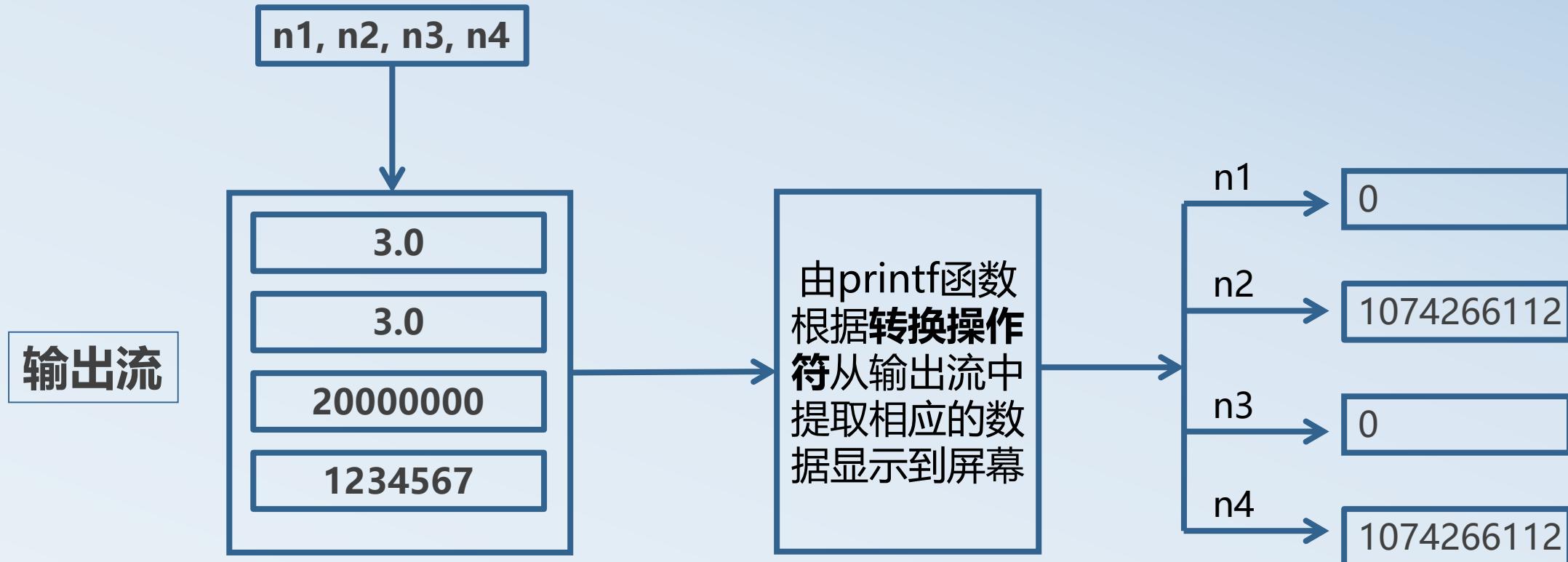
int main(void)
{
    float n1 = 3.0;
    double n2 = 3.0;
    long n3 = 20000000, n4 = 1234567;

    printf("%.1e %.1e %.1e %.1e\n", n1, n2, n3, n4);
    printf("%ld %ld\n", n3, n4);
    printf("%ld %ld %ld %ld\n", n1, n2, n3, n4);

    return 0;
}
```

printf函数的工作过程

```
printf("%ld %ld %ld %ld\n", n1, n2, n3, n4);
```



宽度说明

◆ 最小宽度说明用于指定**显示的最小宽度**

- 当转换值的字符数（含前缀）**小于**最小宽度说明时，则使用填充符将数值填充到最小宽度。
- 当转换值的字符数（含前缀）**大于**最小宽度说明时，最小宽度说明失效。

◆ 例 设int x = 45, y = -4567,

请分析printf("%9d, %4d" , x, y); 语句的输出结果。

◆ 补充：

最小宽度还可以使用*号，然后给出参数值

```
printf("%*d", 5, x);
```

精度说明

◆ 形式: .digit(s)

例: printf("%.**3**f" ,8.1234);

- 对于整数转换，“精度说明”指出要输出的**最少位数**，如果少于最少位数，**可以用0补齐**。
- 当转换操作符为e、E、f时，“精度说明”指出**小数点**后面的数字位数。
- 当转换操作符为g、G时，“精度说明”指出**有效位数**。
- 当转换操作符为 s 时，“精度说明”指出要从字符串输出的**最大字符数**。

printf格式输出

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

int main(void)
{
    int x = 31;
    float f = 30.45;
    char a[10] = "abcd";
    printf("%7.3d\n", x);
    printf("%7.3x\n", x);
    printf("%7.3o\n", x);
    printf("%7.3e\n", f);
    printf("%7.3f\n", f);
    printf("%7.3s\n", a);

    return 0;
}
```

031
01f
037
3.045e+001
30.450
abc
请按任意键继续. . .

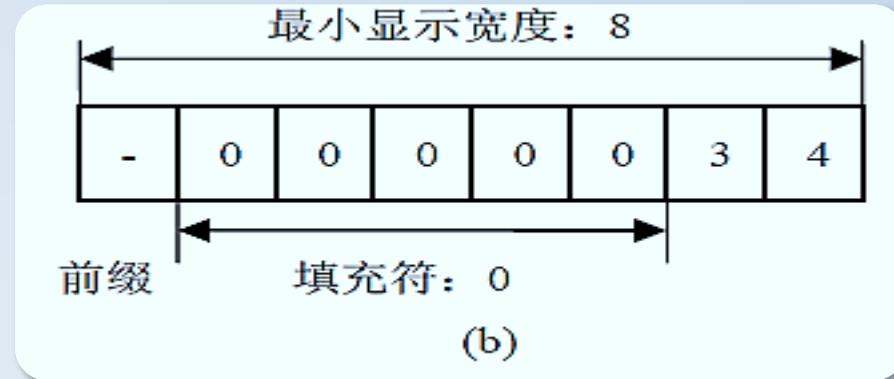
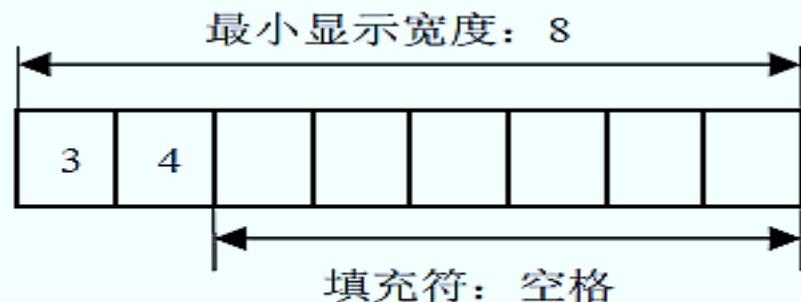
标识字符(I)

❖ 标识字符的作用

标识字符	作用	示例
-	左对齐	"%-20s"
0	如果输出的数值，当输出长度小于字段宽度时，用0填充，遇-无效	"%010d"
+	显示数值的符号，正数显+号，负数显-号	"%+d"
空格	输出的数是正数，显示前导空格，是负数显示-号	"% 10.3f"
#	如果输出的是负数，显示-号 输出的是八进制数，显示前导0 输出是十六进制数，显示前导0x 对于%g来说，可以防止尾随0被删除	"%#d" " %#o" " %#x" " %#g"

标识字符(II)

```
printf("x = %-8d, y = %08d\n", x, y);
```



printf用法提示

❖ 关于宽度和精度说明符 中*

格式	结果	说明
printf("%0*d",6,1234);	001234	*将用6代替
printf("%.*f\n",7,3.233.6782);	233.678	宽度为7, 精度为3

❖ 两个格式操作符中间要有空白字符

格式	结果
printf("%d%f",6,1.234);	61.234
printf("%d %f",6,1.234);	6 1.234

空白字符包括：空格、回车、TAB

printf练一练

❖以下语句的打印结果是？

- int i=79; printf("%o",i);
- float x=333.1234567890; printf("%.2f",x);
- int i=79; printf("%x",i);
- double y=333.1234567890; printf("%2.5f",y);
- int i=7900; printf("%2d",i);
- float x=1.23456789; printf("%.5f",x);

scanf格式输入

❖ 格式输入函数

- 函数作用：
 - 键盘上输入的数据“送到”内存中进行存储

- 一般格式：

scanf(字符串字面值, 参数1, 参数2, ..., 参数n);

- 参数必须是**变量的地址**
- 字符串字面值里除了转换说明符外，**其它字符原样输入**
- 返回值 **成功返回输入到内存的数据个数，失败返回0**
- scanf函数支持1到n个变量的输入：
 - `scanf("%d" , &x);` //1个变量
 - `scanf("x=%d,y=%f" , &x, &y);` //2个参数

scanf格式输入

❖例：从键盘上接收两个整数、一个浮点数，分别存于x、y和z中。

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
    int x, y;
    float z;
    printf("请输入x,y,z 的值（以逗号隔开）:");
    scanf("%d,%d,%f", &x, &y, &z);
    printf("你输入的数是: x=%d,y=%d,z=%f\n", x, y, z);

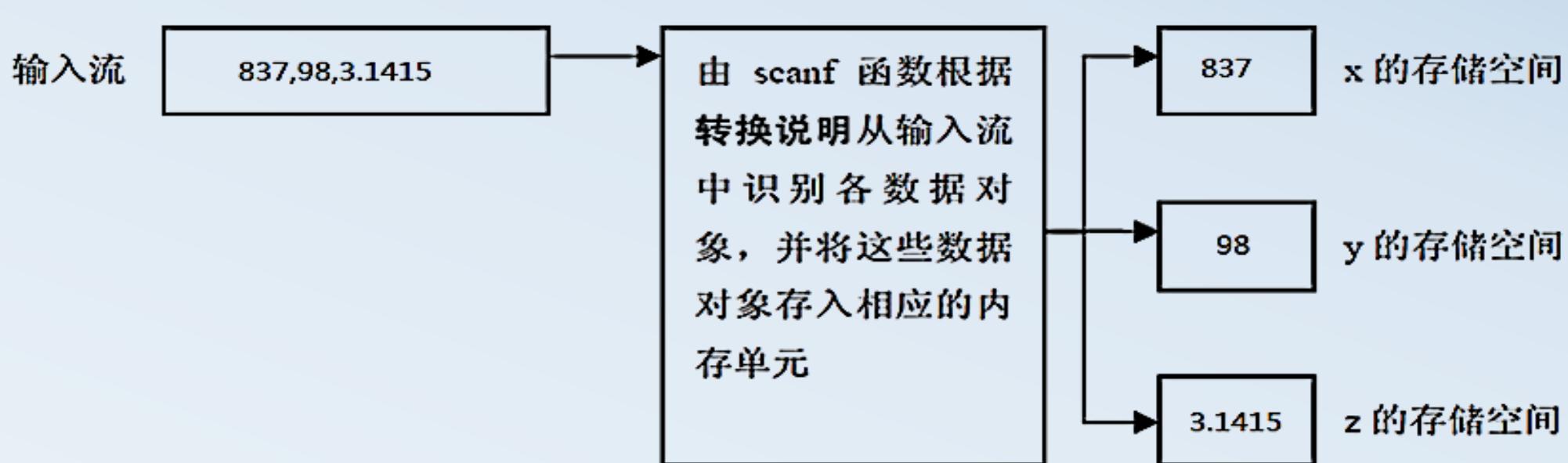
    return 0;
}
```

scanf格式输入

```
scanf("%d,%d,%f", &x, &y, &z);
```

输入流

```
请输入x,y,z 的值（以逗号隔开）:837,98,3.1415  
你输入的数是：x=837,y=98,z=3.141500  
请按任意键继续. . .
```



scanf格式输入

❖一般格式：

- `scanf(字符串字面值, 参数1, 参数2, ..., 参数n);`
- 允许出现在控制字符串中的内容包括：
 - 以%开始的转换说明、转义字符、其他单个字符（如：空格、逗号等）。
- 转换说明一般形式

控制字符串

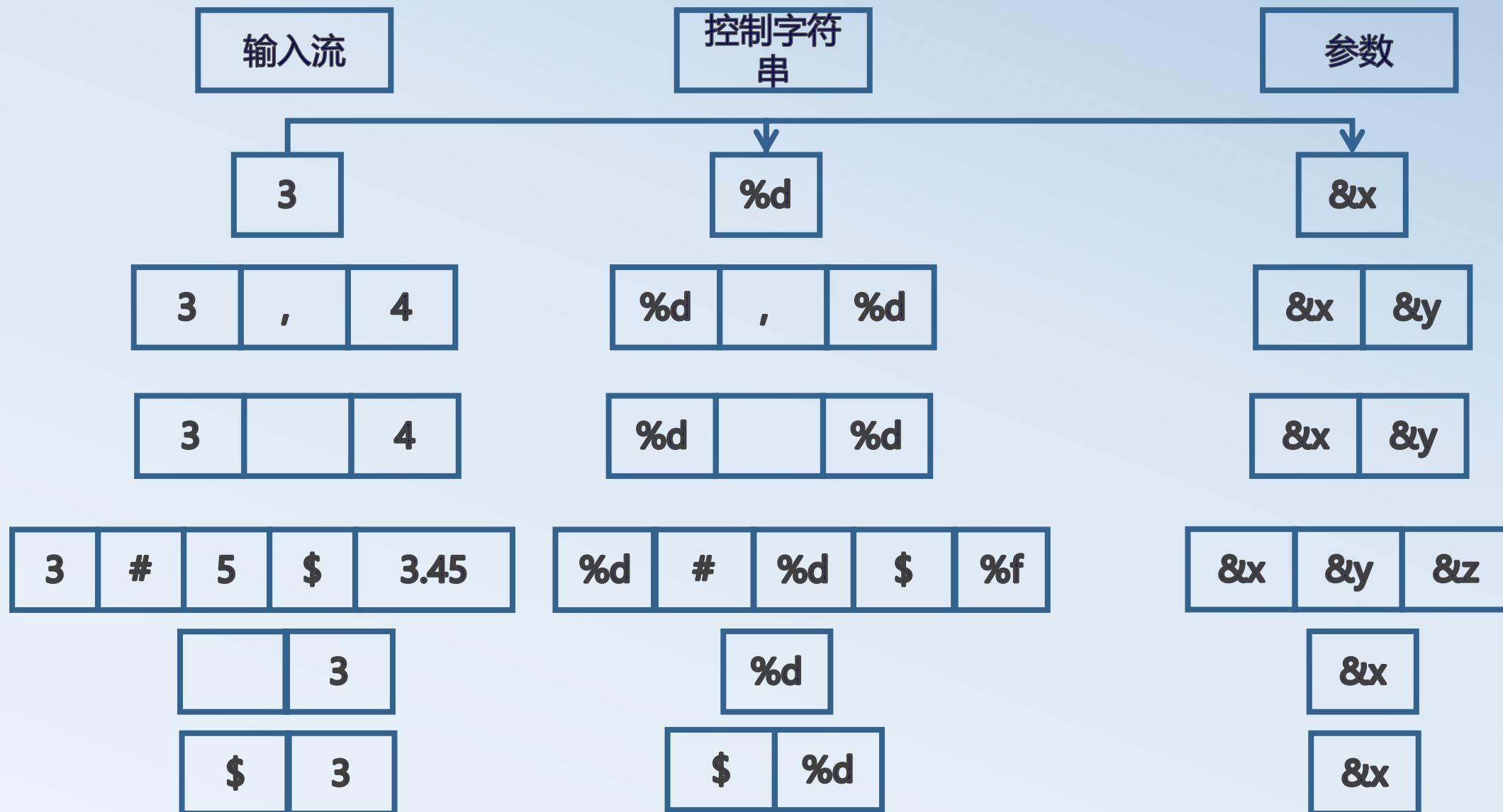
scanf转换操作符

数据类型	转换操作符	含义
int	d、i	输入整数
	u	输入整数
	o	按八进制输入整数
	x、X	按十六进制输入整数
float	f、F	对浮点数按十进制计数法输入
	e、E	对浮点数按科学计数法输入
	g、G	对浮点数按十进制计数法或科学计数法输入
	a、A	浮点数、十六进制数字和p-记数法 (c99)
char	c	输入一个字符
字符串	s	输入一个字符串
其它	%	输入一个百分号
	n	将输入流里当前的字符个数输入到一个整数里，要求操作数为有符号数的地址(了解)

scanf长度修正符

数据类型	转换操作符	含义
short	hd、hi、hu、ho、hx、hX	对short进行格式转换
long	ld、li、lu、lo、lx、lX	对long进行格式转换
long long	lld、lli、llu、llo llx、lX	对long long进行格式转换
double	lf、lf、le、le lg、lg、la、la	把输入的字符解释成double类型的数据
long double	Lf、LF、Le、LE Lg、LG、La、LA	对long double输入
char	hhd, hhi, hhu, hho, hhx, hhX	将输入整数格式转换

scanf格式输入



scanf输入

❖最大宽度说明

- 当某项输入达到最大宽度说明的时候，结束本项输入

❖赋值屏蔽符 (*)

- 如果转换操作符前有*则该项输入不赋值给相应的参数

格式	输入	结果
scanf("%2d%2d" , &a, &b);	1234↙	a=12 b = 34
scanf("%2d%*2d%2d", &a, &b);	123456↙	a=12 b = 56

❖用scanf输入数据时，遇到**空白字符**（字符类型除外）、到达**指定宽度**、**非法输入**，认为该数据数据结束

字符数据的输入输出

❖字符输出函数

- 一般形式: putchar (c)
- 函数作用: 向终端输出一个字符

字符型变量或整型变量

❖字符输入函数

- 一般形式: getchar ()
- 函数作用: 从终端(或系统隐含指定的输入设备)输入一个字符。
- 函数值: 从输入设备得到的字符。

练习

要求

输入三个字母，然后输出这三个字母

目的

学习getchar函数和putchar函数的使用方法

练习

```
#include <stdio.h>
int main(void)
{
    char a,b,c;
    a = getchar();
    b = getchar();
    c = getchar();
    putchar(a);
    putchar(b);
    putchar(c);
    putchar('\n');
    return 0;
}
```

abc 回车
abc

A 回车
B 回车
A
B

本章小结

- ❖ 讲述了实际问题向计算机程序转化的过程。
- ❖ 讲述了数据在计算机内部的存储形式。
- ❖ 讲述了数据类型的种类和区别。
- ❖ 讲述了变量的声明和命名规则。
- ❖ 讲述了常量和字面值的区别。重点理解字面值是有类型的。
- ❖ 讲述了输入及输出函数的用法。

Thank You !