北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

AI 计算机学院

《人工智能的数据基础》

——基本数据表示

主讲教师：冯恺宇

目录

CONTEXT

不同形式的数据在计算机中怎样表示？

在同一计算机中，数据的长度常常是统一的，不足的部分用“0”填充。数据长度以二进制位的多少来统计，但必须是字节（1个字节是8个二进制位）的整数倍数计算

- 正、负数如何表示？
- 小数如何表示？
- 字符、汉字、声音、图形、图像如何表示？

数值型数据

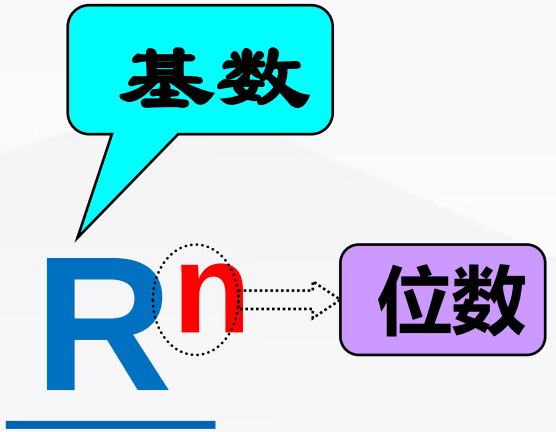
字符型数据

整数的计算机表示

机器数：数（连同符号）在机器中的编码表示

真值：机器数所对应的十进制数值

模数：一个计量器的容量



二进制数值表示与计算

对模数的理解

1. 计数器从“0”开始计数
2. 计数器所能计的数值的个数即模数
3. 计数器的模数 = 最大值+1
4. 计数器的模数（Rn）取决于基数（R）和位数（n）

例题

1. 二位十进制计数器的模数是多少？

基数 ⇒ 10；

位数 ⇒ 2；

模数 ⇒ $10^2 = 100 = 99+1$

2. 八位二进制计数器的模数是多少？

解答

基数 ⇒ 2；

位数 ⇒ 8；

模数 ⇒ $2^8 = 256 = 255+1$

二进制数值表示与计算

- 无符号数
  - 计数 计数时，不需要负数
  - 地址 指向另一个存储单元的地址，不需要负数
- 有符号数怎么表示？

原码

定义：分别用0和1代替数的正号和负号，并置于最高有效位上，绝对值部分置于右端，中间若有空位填上零.

原码的表示范围： $-(2^{n-1}-1)\sim(2^{n-1}-1)$

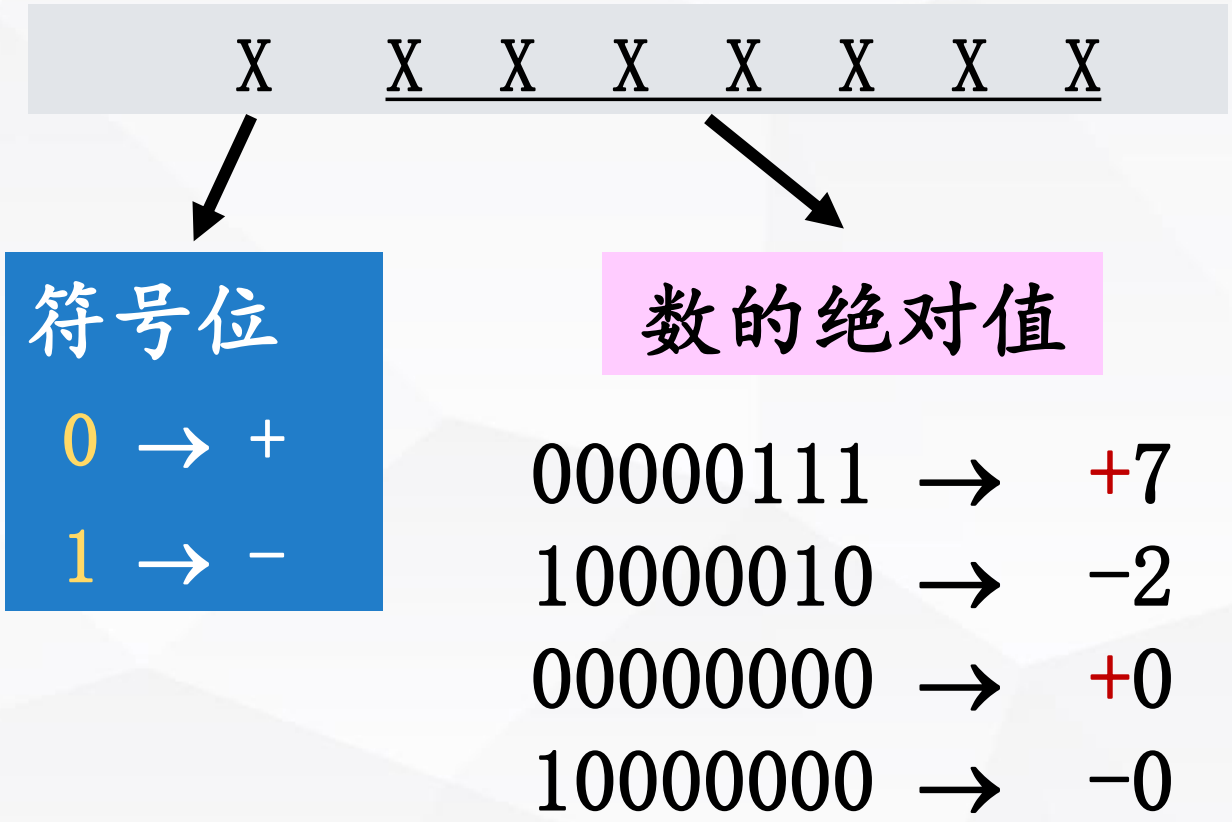
当**n=8**时，原码的表示范围 **-127~+127**

不便于计算

[+0]原 = 0 0000000

[−0]原 = 1 0000000

数的原码表示


X X X X X X X X
符号位
0 → +
1 → −
数的绝对值
00000111 → +7
10000010 → −2
00000000 → +0
10000000 → −0

当n=8时，原码的表示范围是：

1 1 1 1 1 1 1 1 → 0 1 1 1 1 1 1 1

-127 ～ +127

机器数

真值

当n=16时，原码的表示范围是：

1 1 1 ..... 1 1 1 → 0 1 1 ..... 1 1 1

-32767 ～ +32767

n位二进制原码的表示范围：$-(2^{n-1}-1)\sim(2^{n-1}-1)$

最小的负数～最大的正数

反码

定义： 正数的反码表示与其原码表示相同，负数的反码表示是把原码除符号位以外的各位取反。

反码的表示范围： $-(2^{n-1}-1)\sim(2^{n-1}-1)$

便于进行减法等运算

[+0]反=0 0000000
[−0]反=1 1111111

补码

将时针由8点校对到3点

蓝：顺时针 ⇨ +7　　8+7=15 = 3

黄：逆时针 ⇨ −5　　8−5=3

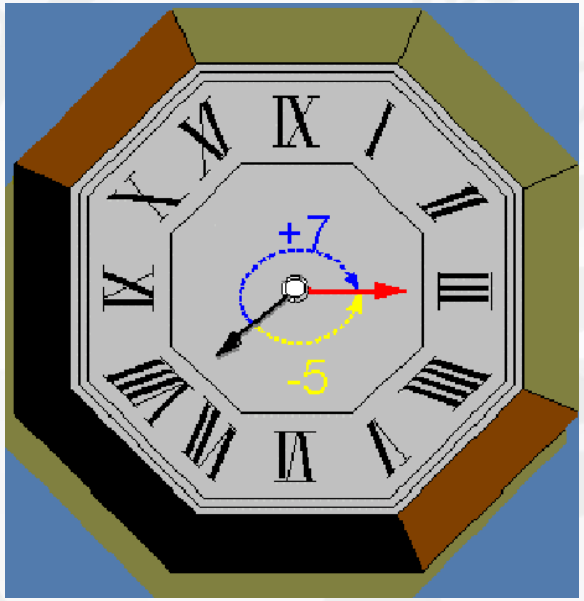
丢掉模数12

$$8 + 7 = 8 - 5 = 3$$

+7是−5的补码

什么是补码？

“0”



二进制数值表示与计算

定义：

$$[X]_{补}=\begin{cases} X & (0\le X<2^{n-1}) \\ 2^{n}-|X| & (-2^{n-1}\le X<0) \end{cases}$$

$$[+0]_{补}=[-0]_{补}=0$$

$[0]_{补}$的两种表示是什么？

补码的表示范围 $-2^{n-1}\sim(2^{n-1}-1)$

当n=8时，补码的表示范围是？

-128 ~ +127

二进制数值表示与计算

补码的算法

1 按定义

$(10000111)_{补}$ =100000000 - 00000111
=11111001

2 原码除符号位外全取反，再加1

-7的原码 **10000111** 取反 **11111000**
加1 **11111001**

原码除符号位外全取反，直到最后一个1为止

-7的原码 **10000111** -8的原码 **10001000**

补码 **11111001** 补码 **11111000**

为什么负数补码的编码多一个？

讨论

当n=4时，-2^3 的补码是？

$$[-2^3]_{补} = 2^4 - |-2^3|$$

$$= 2^4 - 2^3$$

$$= 2^3$$

$$= 1000$$

当n=4时，补码的表示范围是：-8 ~ +7

十进制	原码	补码
-0	1 0 0 0	0 0 0 0
-1	1 0 0 1	1 1 1 1
-2	1 0 1 0	1 1 1 0
-3	1 0 1 1	1 1 0 1
-4	1 1 0 0	1 1 0 0
-5	1 1 0 1	1 0 1 1
-6	1 1 1 0	1 0 1 0
-7	1 1 1 1	1 0 0 1
-8		1 0 0 0

将补码的机器数看作模16，则-1的补码是15，运算时连同符号位一起参加运算

当n=8时，补码的表示范围是：

1 0 0 0 0 0 0 0 → 0 1 1 1 1 1 1 1

-128 ～ +127

当n=16时，补码的表示范围是：

1 0 0…… 0 0 → 0 1 1 …… 1 1 1

-32768 ～ +32767

真值

机器数

n位二进制补码的表示范围：$-2^{n-1} \sim (2^{n-1}-1)$

最小的负数～最大的正数

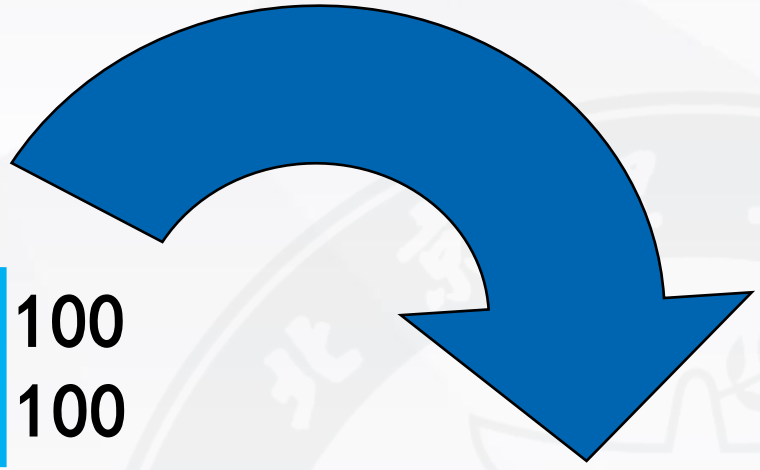
二进制数值表示与计算

为什么用补码表示？

3位2进制数：

111−011= 100

111+(1000−011)=111+101= 1 100



（2^3 －3）是−3的补码

用补码表示
可用加法代替减法

补码运算

计算机系统通常采用补码运算；

仅用加法器就可实现所有算术运算；

符号位和数值部分一样参加运算。

根据补码的定义，对于$[a]_{补}+[b]_{补}$有规则：

$$[a]_{补}+[b]_{补}=2^n+a+2^n+b$$
$$=2^n+(a+b) \quad \text{丢掉一个模}2^n$$
$$=[a+b]_{补}$$

同理：

$$[a]_{补}-[b]_{补}=[a-b]_{补}$$

习题

用8位二进制写出0和-128的原码和补码

解答

- $[0]_{补}=[0]_{原}=00000000$
- $[-128]_{原}$ 无法表示
- $[-128]_{补}=2^8-128=2^7=128=10000000$

C语言整型数据类型及表示

整数类型

基本整型
短整型
长整型
无符号整型
无符号短整型
无符号长整型

```
Example:
main()
{ int a;
   a=10;
   printf("%d",a);
}
```

问题：为什么要多种？
每一种是否都必须用？

C语言整型数据类型及表示

整型变量的形式

	约定规则	占内存字节数
共六种	基本整型 int	2
	短整型 short int	2
	长整型 long int	4
	无符号整型 unsigned int	2
	无符号短整型 unsigned short	2
	无符号长整型 unsigned long	4

占存储空间

整型变量的定义

格式：类型说明符　变量列表；

例：
```
int   i, j;
long  k, m;
unsigned int x,y
```

变量有值吗?

C中的所有变量必须先定义后使用！！

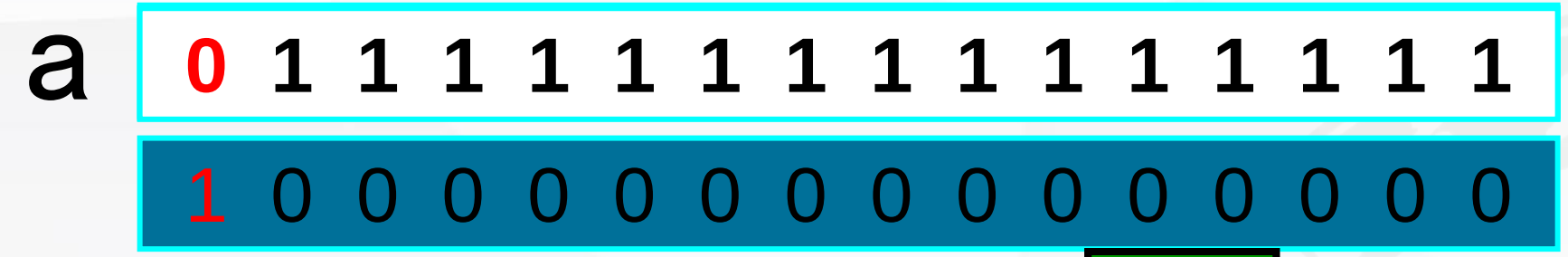
C语言整型数据类型及表示

j、m、x 所占存储容量和取值范围？

类型	类型说明符	长度	数的范围
基本型	int	2字节	-32768～32767
短整型	short	2字节	-2^{15}～$2^{15}-1$
长整型	long	4字节	-2^{31}～$2^{31}-1$
无符号整型	unsigned	2字节	0～65535
无符号短整型	unsigned short	2字节	0～65535
无符号长整型	unsigned long	4字节	0～（$2^{32}-1$）

Sizeof（）

int型数的表示范围：存储长度16位二进制(2B)



最大的正数？ $=2^{15}-1=32767$

最大的负数？ $=-2^{15}=-32768$

- 定点二进制补码表示
- 占2个字节存储
- 数的表示范围：-32768～32767

整型数据应用中的几个问题

- 变量要先定义后使用；
- 数据溢出；
- 常量与变量的类型要匹配。

例：编写求两数和的C程序并上机运行，程序如下：

例3-1 /*求两数和的C程序*/

```
main()                          /* 求两数和主函数 */
 {
    int a,b,c;                  /* 说明a、b为整型变量 */
    a=32767;                    /* 为变量a赋最大值 */
    b=3;                        /* 为变量b赋值 */
    c=a+b;     /* 计算a+b并将结果赋值给变量c */
   printf("c=%d\n",c);          /* 输出变量c的值 */
 }
```

整型数据表示

- 整型常量的类型
  根据值大小默认类型
  在常量后面加l或L，表示long int型
  例如：123L，0L，432l 都是long int型常量

- 整型常量的不同数值表示方法
  10进制：0~9
  8进制：0~7，以0开头
  16进制：0~9，A~F/a~f，以0x或0X开头

  100、-8、0；010、024；0x18、0X1F

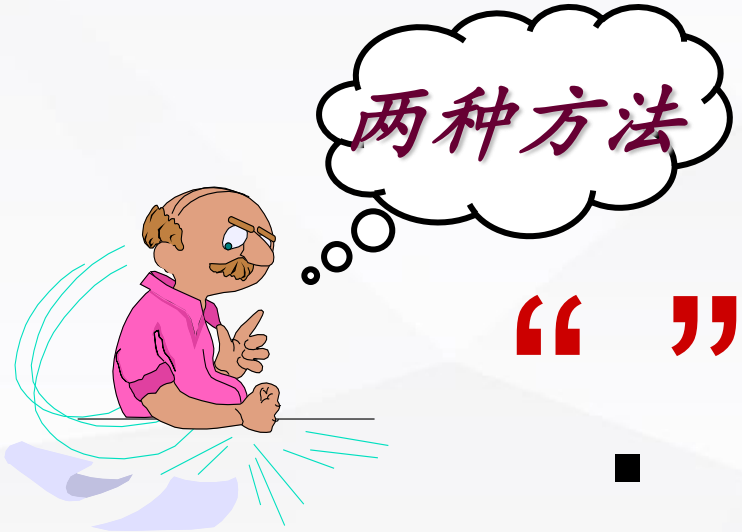
整型数据包括整型常量和整型变量，整型数据以二进制补码形式存储

实数的计算机表示

位置固定：3.14159--定点

位置变化：3.14159--浮点

小数点在计算机中如何表示



“ ”

- 的位置移动，数的大小不变

定点表示法：小数点位置固定在某一位置

包括：定点小数和定点整数

定点小数格式

小数点固定在最高数据位的左边

所有的数都是小于1的纯小数

。。。

如出现大于或等于1的情况，定点小数格式就无法正确地表示出来，这种情况称为“溢出”

实数的计算机表示

当：X=123456时

比例因子：　k=1000000

X’=x/k

=123456/1000000

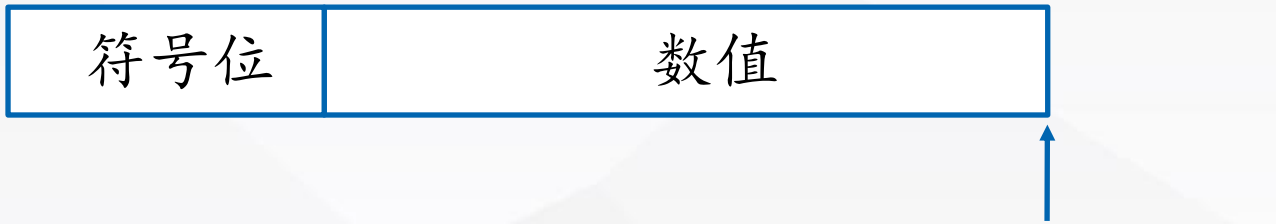
=0.123456

可以用定点小数表示了！

…… 最后的结果再乘上“比例因子”

定点整数格式

小数点固定在最低位数字的右边



是不是除一个比例因子就能表示小数?

定点表示的特点?

直观、简单、节省硬件

数据范围小，不灵活

二进制数值表示与计算

浮点表示法　小数点位置可任意移动

3.14159

$3.14159*10^0$

$0.314159*10^1$

$0.0314159*10^2$

$M \cdot R^E$ → 阶

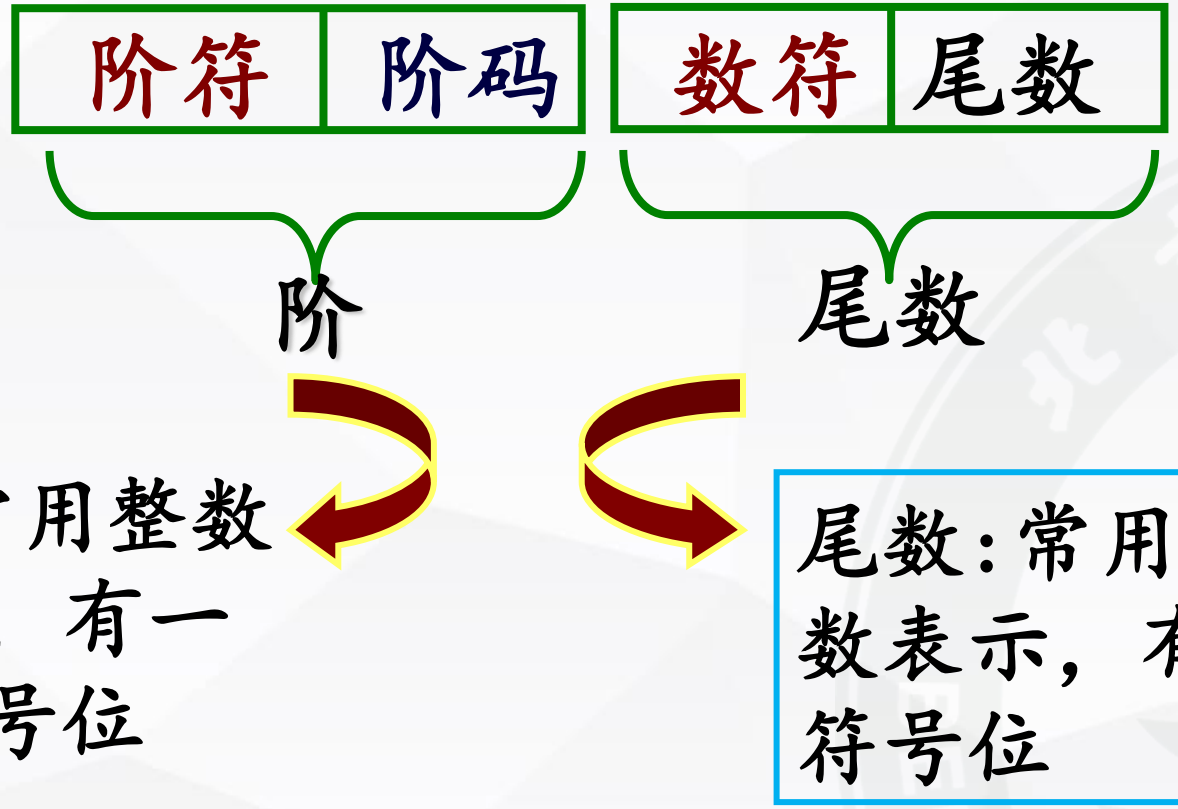
尾数　基数

尾数：数的有效数字

阶：小数在数中的实际位置

浮点数的规格化：　非零浮点数的尾数最高位必须是1

浮点表示法的特点：


阶符
阶码
数符
尾数
阶
尾数
阶：常用整数表示，有一个符号位
尾数：常用定点小数表示，有一个符号位

二进制数值表示与计算

计算机中通常表示浮点数的字长为16位，用8位作阶，含一位阶符，8位作尾数，含一位数符

$$
\begin{aligned}
&(72.45\times10^{5})_{10}\\
=\ &(11011101000110011001000)_{2}\\
\approx\ &(0.1101110)_{2}\times2^{23}\\
=\ &(0.1101110)_{2}\times2^{(10111)_2}
\end{aligned}
$$

0	0010111	0	1101110

二进制数值表示与计算

假定：32位字长，8位作阶，24位作尾数

能表示的最大的数？最小数？

当阶的符号位为 0，其余为 1
尾数符号位为 0，其余为 1 时，表示的数最大：10^{38}

当阶的符号位为 0，其余为 1
尾数符号位为 1，其余为 1 时，表示的数最小：-10^{38}

怎么得到的

$$2^{2^7-1} * (1-2^{-23}) \approx 2^{127} \approx 10^{38}$$

习题 设字长16位，4位阶码，12位尾数

1. 指出所能表示的数的范围，非零的最小绝对值

数的范围： $+2^7$模~ -2^7 (4位-1位符号）

最小绝对值： 2^{-18} (11位+7位）

2. 图示0.00011001在该字中的规格化表示

1011	011001000000

实型变量

阶码：$2^{127}=1.70141\times10^{38}$

实型变量的有关规则如下：

	类型说明符	长度	数的范围	有效数字
单精度型	float	4字节	-10^{38} ~ 10^{38}	7位
双精度型	double型	8字节	-10^{308} ~10^{308}	15位

什么是数的范围？什么是有效位？

2^{-23} <= |尾数| <= $1-2^{-23}$

实型常量 C语言实型数据包括实型常量和实型变量

两种表示形式
- 小数 0.123
- 指数 3e-3

实型常量只能用十进制形式表示，不能用八进制和十六进制。

✓ 1e3、1.8e-3、-123e-6、-.1e-3

✗ e5、1e-3.2、e、.e-03

二进制数值表示与计算

- 单精度实数的精度取决于小数部分的23位二进制数位所能表达的数值位数，将其转换为十进制，最多可表示7位十进制数字，所以单精度实数的**有效位是7位**

- 双精度型用于扩大存储位数，目的是增加实数的长度，减少累积误差，改善计算精度，双精度实数的**有效位是15位**

例3-1：

```
float a,b; /* 说明变量 a,b 为单精度型实数 */
double c,d; /* 说明变量 c,d 为双精度型实数 */
long double e,f; /* 说明变量 e,f 为长双精度型实数 */
```

实型数据应用中的误差问题！！

由于机器存储的限制，使用实型数据会产生一些误差，运算次数愈多，误差积累就愈大，所以要注意实型数据的有效位，合理使用不同的类型，尽可能减少误差。

二进制数值表示与计算

例3-2：输出实型数据a,b

```
main() /*P3-2*/
 {float a; /* 说明变量 a为单精度型 */
  double b; /* 说明变量 b为双精度型 */
  a=12345.6789; /* 为a赋值 */
  b=0.123456789123456789e15; /* 为b赋值 */
  printf("a=%f,b=%f\n",a,b);/* 输出变量a、b */
 }
```

理想结果应该是照原样输出，即：

a=12345.6789，b=0.123456789123456789e15

实际输出结果是：

a=12345.678711，b=123456789123456.797000

为什么？？

超出表示范围的数机器如何处理？

实型数的分辨率

最小正数：10^{-38} 小于该数时处理为0 下溢

最大正数：10^{38} 大于该数时提示出错 上溢

- 实型数在计算机内部都按指数形式存放
- 实型数常数不区分单精度和双精度
- 数的存储结构和精度与计算机硬件特性有关

逻辑运算与计算机控制

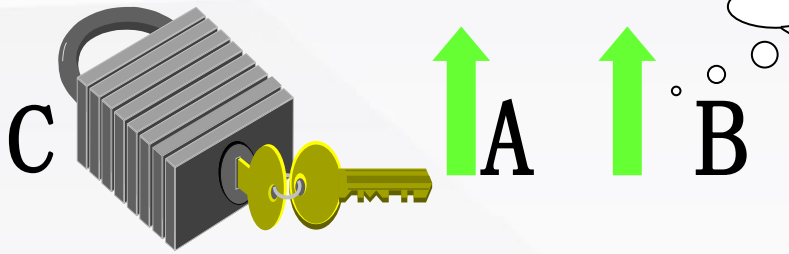
逻辑运算符连接逻辑变量，构成逻辑表达式，结果为逻辑值

逻辑门是计算机硬件电路的基础，是描述数字逻辑电路的最基本单元部件，输入信号经由一定的逻辑门可以得到一定的输出信号

在逻辑门电路中，任何信号只存在两种状态，即高电平和低电平，对应到逻辑运算，以高电平来表示逻辑“1”（真）、以低电平来表示逻辑“0”（假）

二进制数值表示与计算

逻辑或(加、+、∪)



两把钥匙

锁打开：1
关：0
有钥匙：1
无：0

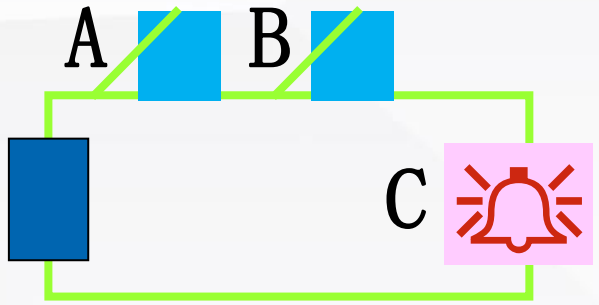
两个条件中只要有一个成立，结果就成立

有真值表：

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

二进制数值表示与计算

逻辑与（乘、×、∩）



两个开关A、B
控制一盏灯C

开关合：1
开：0
灯亮：1
灭：0

两个条件都个成立
结果才成立

有真值表

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

逻辑否定(非、A’、$\overline{A}$)

A=8, B=10　A<B 为1

$\overline{A<B}$ 为 0

有真值表

A	C
0	1
1	0

异或（⊕）两逻辑变量相同时为 0
不同时为 1

有真值表

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

注意！

按位运算先非后与再或（异或）

例题

已知：A = 0, B = 1, C = 1

求：$Z = (A \cap \overline{B} \cap C) \cup (A \cap \overline{B}) \cap (\overline{B} \vee C)$

解答

$$Z = (0 \cap \overline{1} \cap 1) \cup (0 \cap \overline{1}) \cap (\overline{1} \vee 1)$$
$$= (0 \cap 0 \cap 1) \cup (1 \cap 1) \cap (1 \vee 0)$$
$$= (0) \cup (1) \cap (1)$$
$$= 0 \cup 1 = 1$$

二进制数值表示与计算

习题

已知：A=01001101　B=10011001

求：A∩B，A∪B，A⊕B，$\overline{A}$ 的结果

解答

A∩B=00001001

01001101

∩ 10011001

00001001

A∪B=11011101

01001101

∪ 10011001

11011101

A⊕B=11010100

01001101

⊕ 10011001

11010100

$\overline{A}$=11010100

01001101

10110010

二进制数值表示与计算

C语言逻辑变量

无（借用其他变量）

C语言逻辑值

0为逻辑假（0）

非零为逻辑真（1）

ASCII码

America Standard Code for Information Interchange

美国标准信息交换码，是目前国际上最为流行的字符信息编码方案

这套字符集共有128个（00～7FH），其中包括26个英文字母的大小写符号编码以及一些标点符号、专用符号及控制符（如回车、换行、响铃等），ASCII用7位二进制编码，恰好可以表示128种字符和控制符号。

西文字符的编码

ASCII码表

b6b5b4 / b3b2b1b0	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	'	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	EXT	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	（	8	H	X	h	x
1001	HT	EM	）	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	−	=	M	]	m	}
1110	SO	RS	.	>	N	↑	n	～
1111	SI	US	/	?	O	←	o	DEL

ASCII中的特殊控制符的意义或动作

符号	意义或动作	符号	意义或动作	符号	意义或动作
NUL	空	FF	走纸控制	ETB	信息组传送束
SOH	标题开始	CR	回车	CAN	作废
STX	正文开始	SO	移位输出	EM	纸尽
EXT	正文结束	SI	移位输入	SUB	减
EOT	传输结束	SP	空格	ESC	换码
ENQ	询问	DLE	数据链换码	FS	文字分隔符
ACK	承认	DC1	设备控制1	GS	组分隔符
BEL	响铃报警	DC2	设备控制2	RS	记录分隔符
BS	退一格	DC3	设备控制3	US	单元分隔符
HT	横向列表	DC4	设备控制4	DEL	删除
LF	换行	NAK	否定		
VT	垂直列表	SYN	空转同步		

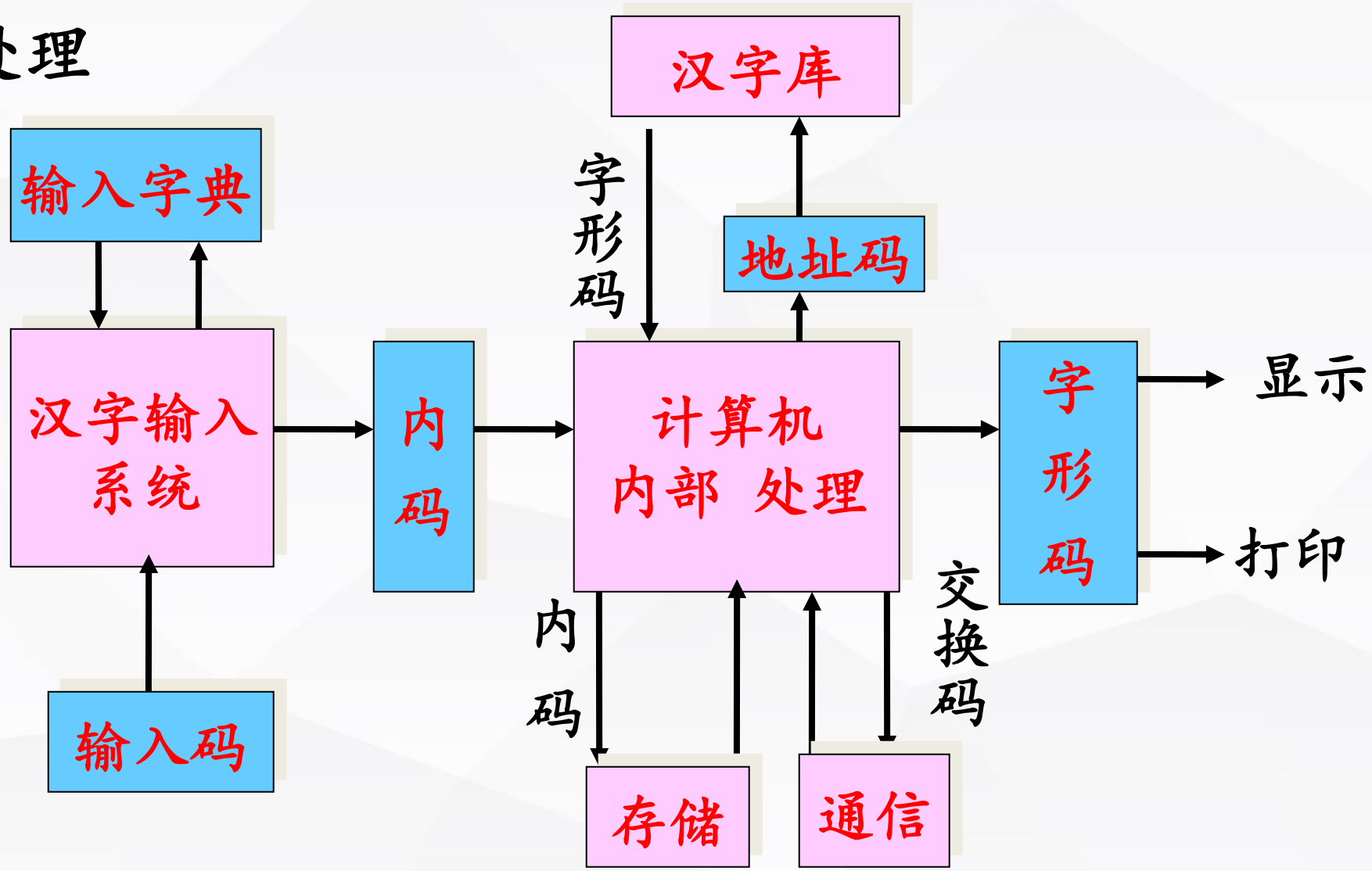
汉字信息的编码

汉字的特点

1. 字数多
   共6万左右，需要的编码多
2. 字形复杂
   20画以上需要字模点阵
3. 同音字多
   需要输入方法灵活

计算机对汉字的处理就是对汉字代码进行转换

汉字信息处理



1. 汉字输入码

汉字输入码是指在键盘上利用数字、符号或拼音字母将汉字以代码的形式输入的代码.

2. 汉字国标码

汉字国标码是我国1981年公布的“中华人民共和国国家标准信息交换汉字编码（GB2312—80）”代码。国标码中有6763个汉字和682个其他基本图形字符，共计7445个字符。

3. 汉字机内码

汉字机内码是指一个汉字被计算机系统内部处理和存储而使用的代码。

4. 汉字字形码

汉字字形码又称汉字字模，它是指一个汉字供显示器和打印机输出的字形点阵代码。

UNICODE码(国际码)

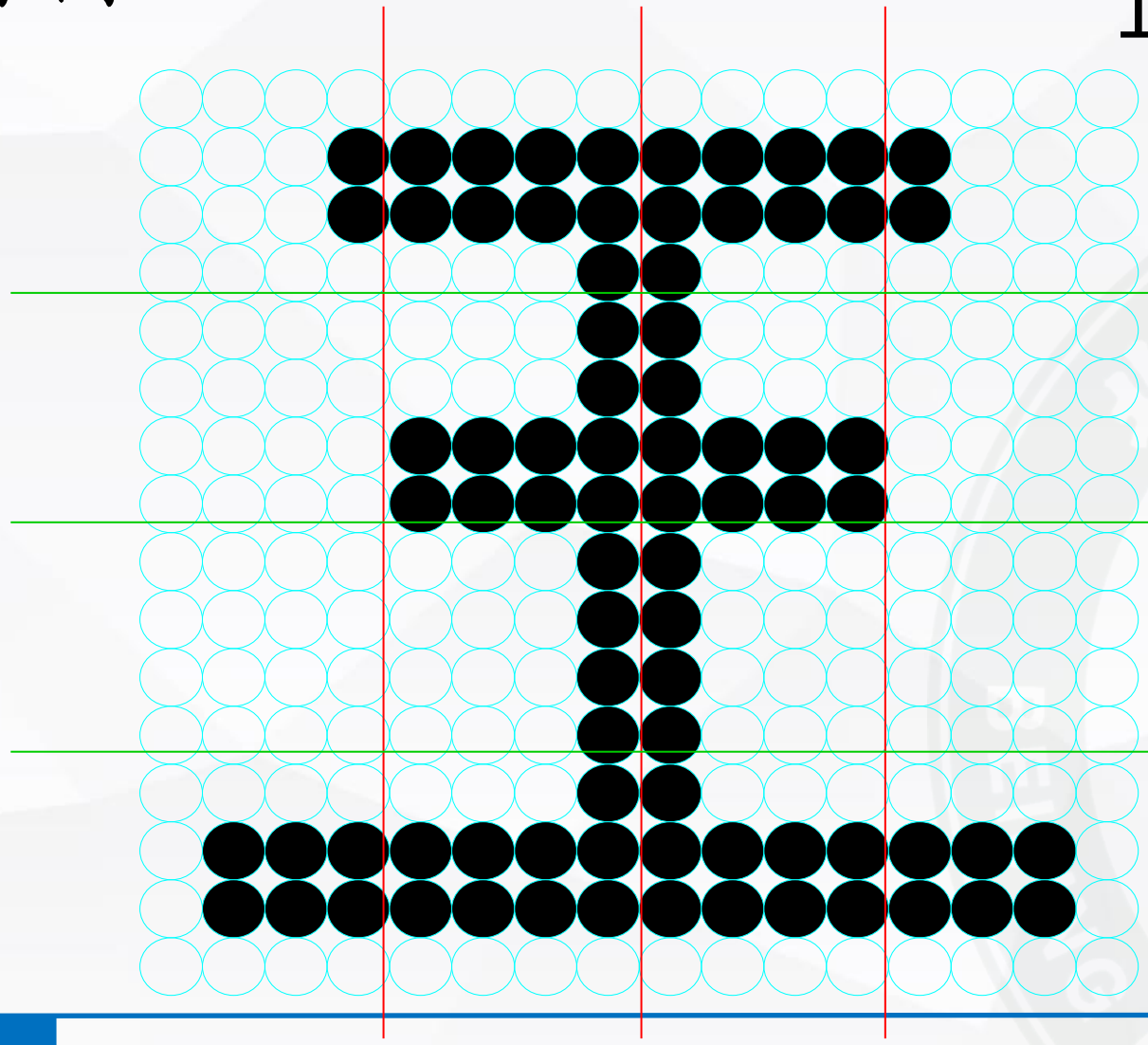
用16位2进制位表示，可以表示65536个字符

UNICODE码

可以表达世界各种语言，还保留30000多个编码供将来使用。UNICODE的前256个编码与ASCII码相同。

字符信息编码与标准交换

字形码示例



16×16点阵

0 0 0 0
1 F F 8
1 F F 8
0 1 8 0
0 1 8 0
0 1 8 0
0 F F 0
0 F F 0
0 1 8 0
0 1 8 0
0 1 8 0
0 1 8 0
0 1 8 0
7 F F E
7 F F E
0 0 0 0

汉字字形码

汉字点阵类型	点阵	占用字节数
简易型	16×16	32
普及型	24×24	72
提高型	32×32	128
精密型	96×96	1152

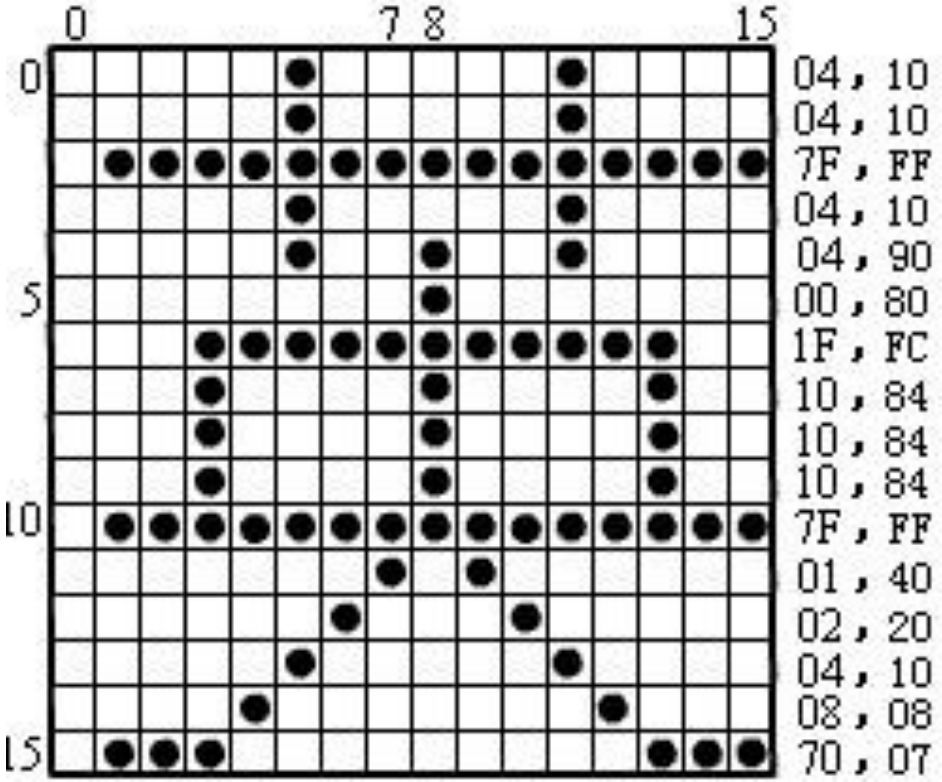
点阵数越高字型质量越好，但占用存储空间越大。

精密型汉字字形通常采用信息压缩存储技术。

当需要输出某个汉字时，将该汉字的字模找出，输出设备按照字模进行输出。

[情景问题2-3]

每到节日的夜晚，我们会看到大街上各种霓虹灯构成的流光溢彩，例如：国庆节的夜晚，学校门口高高悬空的耀眼的“国庆”两个字，字的每一个笔画处都持续转动着红色流动串。这是用什么方法实现的？和二进制编码有什么关系？色彩是怎么转动起来的？什么原理？



通用字符编码集

- 相同的编码在不同的字符集中可能有不同的意义。
- 国际标准 ISO 10646 定义了一种通用字符集UCS（Universal Character Set），它包含了世界上大多数可书写的字符系统。

[练习与思考2-3] P31

C语言字符型数据及表示 — 字符数字化

字型符号 → 编码

字符型数据的存储与计算

字符“A”的ASCII码的二进制表示

十进制整型数65的二进制表示

0	1	0	0	0	0	0	1

例：“A”的存储为：

0	1	0	0	0	0	0	1

0 1 0 0 0 0 0 1

字符信息编码与标准交换

字符“A”的存储形式是一个整型数65，但占一个字节，而整型数65占两个字节，可以进行算术运算、混合运算，可以与整型变量相互赋值，也可以将字符型数据以字符或整数两种形式输出。

十进制整型数321
输出为字符是?

0 0 0 0 0 0 0 1	0 1 0 0 0 0 0 1

字符‘a’和一个整型数是这样运算的

01100001

00000001 01000001

在ASCII范围以内，整型数据与字符型数据可以通用

转义字符

转义字符是特殊形式的字符常量

- 转义字符的概念：是一种特殊的字符常量，用于表示常用的但却难以用一般形式表示的不可显示字符。

- 转义字符的表示：用一个转义标识符“\”开头，后面是需要的转义字符。常用的转义字符序列的字符常量见表：

转义字符表

转义字符	功能
\n	换行
\t	水平跳格
\b	退格
\r	回车
\f	走纸换页
\\	反斜线字符
\'	单引号字符
\“	双引号字符
\ddd	1至3位八进制数表示的字符
\xdd	1至2位十六进制数表示的字符
\0	NULL

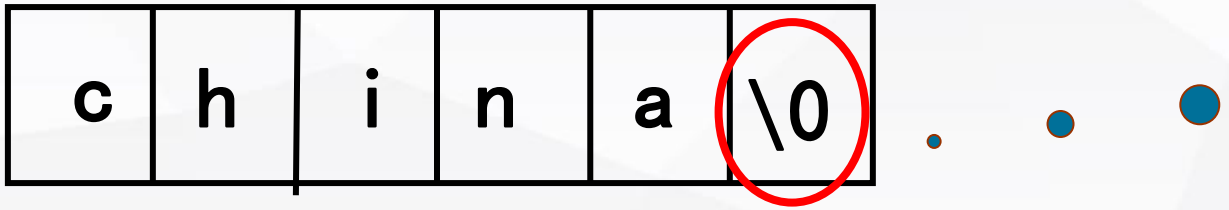
转义符“\”表示将其后的字符原来的含义进行转换，变成某种另外特殊约定的含义。

字符串常量

“a”		串长 1
“This is C string”		串长16
“ ”	（空格）	串长1
“”	（不含空格）	串长0

C 中没有专用的字符串变量

n个字符组成的字符串常量，占空间为 n+1个字节



字符串结束标记
“空”字符

字符型常量

字符型常量的值是该字符对应的 ASCII 码值

Example:

value of 'A' is 65 (in ASCII set)
value of '0' is 48 (in ASCII set), ***not zero***
value of '(' is 40 (in ASCII set)

字符型数据可以参与数值运算，常常做字符比较运算

Example: ***char a = '9', b= '0'; int c;***
c = a –b;

c= 9