中南大學
CENTRAL SOUTH UNIVERSITY

第七讲
基本类型

7、基本类型

7.1 整数类型

- **有**符号数、**无**符号数
- 短整型、整形、长整型
- 组合六种

```
short int          unsigned short int
int                unsigned int
long int           unsigned long int
```

- C程序员经常会省略int

Typical ranges on a 32-bit machine

Type	*Smallest Value*	*Largest Value*
short int	–32,768	32,767
unsigned short int	0	65,535
int	–2,147,483,648	2,147,483,647
unsigned int	0	4,294,967,295
long int	–2,147,483,648	2,147,483,647
unsigned long int	0	4,294,967,295

7.1 整数类型

- C99中
  - `long long int` and `unsigned long long int`
  - 至少64位宽
- 整数溢出
  - 结果不可预见

```
x=2147483647;
y=1;
printf("%d",x+y);
```

```
-2147483648
```

```
x=2147483647;
y=2;
printf("%d",x+y);
```

```
-2147483647
```

7、基本类型

7.1 整数类型

- 读写整数
  - %u： 无符号数
  - %o： 八进制
  - %x： 十六进制
- 读写短整数
  - 在d、o、u前加字母h
- 读写长整数
  - 在d、o、u前加字母l
- 读写长长整数
  - 在d、o、u前加字母ll

```
unsigned int u;

scanf("%u", &u);   /* reads  u in base 10 */
printf("%u", u);   /* writes u in base 10 */
scanf("%o", &u);   /* reads  u in base  8 */
printf("%o", u);   /* writes u in base  8 */
scanf("%x", &u);   /* reads  u in base 16 */
printf("%x", u);   /* writes u in base 16 */
```

```
short s;

scanf("%hd", &s);
printf("%hd", s);
```

```
long l;

scanf("%ld", &l);
printf("%ld", l);
```

```
long long ll;

scanf("%lld", &ll);
printf("%lld", ll);
```

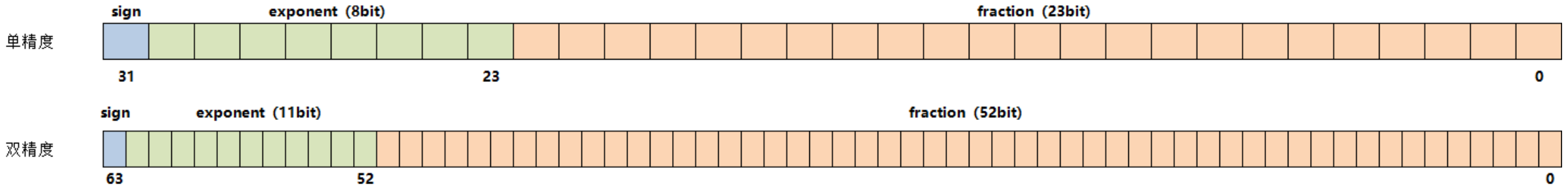
7.1 整数类型

长整形输入输出

```c
/* Sums a series of numbers (using long variables) */

#include <stdio.h>

int main(void)
{
  long n, sum = 0;

  printf("This program sums a series of integers.\n");
  printf("Enter integers (0 to terminate): ");

  scanf("%ld", &n);
  while (n != 0) {
    sum += n;
    scanf("%ld", &n);
  }
  printf("The sum is: %ld\n", sum);

  return 0;
}
```

7.2 浮点类型

- float：**单**精度浮点数
- double：**双**精度浮点数
- long double：扩展精度浮点数
- 标准IEEE754
  - 单精度：4个字节；1个符号位，8个指数位，23个尾数位
  - 双精度：8个字节；1个符号位，11个指数位，52个尾数位
- 范围
  - 单精度最大值约$3.4*10^{38}$
  - 双精度最大值约$1.8*10^{308}$



7.2 浮点类型

- 浮点常量
  - 57.0可以如下表示：
    - 57.0 57. 57.0e0 57E0 5.7e1 5.7e+1 .57e2 570.e-1
- 读写浮点数

变量定义	格式符	输入输出
float x;	%f %e %g	scanf("%f", &x); printf("%f",x);
double y;	%lf %f %e %g	scanf("%lf", &y); printf("%f", y);
long double z;	%Lf	scanf("%Lf", &z); printf("%Lf", z);

7.3 字符类型

- 字符集
  - ASCII

```
char ch;

ch = 'a';    /* lower-case a */
ch = 'A';    /* upper-case A */
ch = '0';    /* zero         */
ch = ' ';    /* space        */
```

  - 字符常量需要用**单引号**括起来，而不是双引号

7、基本类型

7.3 字符类型

ASCII表

（ American Standard Code for Information Interchange 美国标准信息交换代码 ）

高四位 / 低四位		ASCII控制字符												ASCII打印字符											
		0000						0001						0010		0011		0100		0101		0100		0111	
		0						1						2		3		4		5		6		7	
		十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	Ctrl
0000	0	0		^@	NUL	\0	空字符	16	▶	^P	DLE		数据链路转义	32		48	0	64	@	80	P	96	`	112	p	
0001	1	1	☺	^A	SOH		标题开始	17	◀	^Q	DC1		设备控制 1	33	!	49	1	65	A	81	Q	97	a	113	q	
0010	2	2	☻	^B	STX		正文开始	18	↕	^R	DC2		设备控制 2	34	"	50	2	66	B	82	R	98	b	114	r	
0011	3	3	♥	^C	ETX		正文结束	19	‼	^S	DC3		设备控制 3	35	#	51	3	67	C	83	S	99	c	115	s	
0100	4	4	♦	^D	EOT		传输结束	20	¶	^T	DC4		设备控制 4	36	$	52	4	68	D	84	T	100	d	116	t	
0101	5	5	♣	^E	ENQ		查询	21	§	^U	NAK		否定应答	37	%	53	5	69	E	85	U	101	e	117	u	
0110	6	6	♠	^F	ACK		肯定应答	22	▬	^V	SYN		同步空闲	38	&	54	6	70	F	86	V	102	f	118	v	
0111	7	7	•	^G	BEL	\a	响铃	23	↨	^W	ETB		传输块结束	39	'	55	7	71	G	87	W	103	g	119	w	
1000	8	8	◘	^H	BS	\b	退格	24	↑	^X	CAN		取消	40	(	56	8	72	H	88	X	104	h	120	x	
1001	9	9	○	^I	HT	\t	横向制表	25	↓	^Y	EM		介质结束	41	)	57	9	73	I	89	Y	105	i	121	y	
1010	A	10	◙	^J	LF	\n	换行	26	→	^Z	SUB		替代	42	*	58	:	74	J	90	Z	106	j	122	z	
1011	B	11	♂	^K	VT	\v	纵向制表	27	←	^[	ESC	\e	溢出	43	+	59	;	75	K	91	[	107	k	123	{	
1100	C	12	♀	^L	FF	\f	换页	28	∟	^\	FS		文件分隔符	44	,	60	<	76	L	92	\	108	l	124	|	
1101	D	13	♪	^M	CR	\r	回车	29	↔	^]	GS		组分隔符	45	-	61	=	77	M	93	]	109	m	125	}	
1110	E	14	♫	^N	SO		移出	30	▲	^^	RS		记录分隔符	46	.	62	>	78	N	94	^	110	n	126	~	
1111	F	15	☼	^O	SI		移入	31	▼	^-	US		单元分隔符	47	/	63	?	79	O	95	_	111	o	127	⌂	^Backspace 代码：DEL

7.3 字符类型

字符操作

- C语言把字符当作**小整数**进行处理
- 字符‘a’的值97（61H）

```
char ch;
int i;

i = 'a';        /* i is now 97    */
ch = 65;        /* ch is now 'A'  */
ch = ch + 1;    /* ch is now 'B'  */
ch++;           /* ch is now 'C'  */
```

小写字符转换为大写字符

```
if ('a' <= ch && ch <= 'z')
  ch = ch - 'a' + 'A';
```

输出所有大写字母的ASCII码

```
for (ch = 'A'; ch <= 'Z'; ch++)
  printf("%5d", ch);
```

```
65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90
```

7.3 字符类型

转义序列

响铃	\a
回退	\b
翻页	\f
换行	\n
回车	\r
水平制表符	\t
垂直制表符	\v
反斜杠	\\
问号	\?
单引号	\'
双引号	\"

7.3 字符类型

- 用scanf和printf读写字符

```
char ch;

scanf("%c", &ch);   /* reads one character */
printf("%c", ch);   /* writes one character */
```

- 在读入字符前，scanf函数不会跳过空白字符
- 为了强制scanf函数在读入字符前跳过空白字符，需要在格式说明%c前面加一个空格

```
scanf(" %c", &ch);
```

7.3 字符类型

- 用getchar和putchar读写字符
- putchar：写单个字符

```
putchar(ch);
```

- getchar：读入一个字符并返回

```
ch = getchar();
```

```
do {
    scanf("%c", &ch);
} while (ch != '\n');
```

```
do {
    ch = getchar();
} while (ch != '\n');
```

```
while ((ch = getchar()) != '\n’)  ;
```

- 示例：确定消息长度

```
/*07-01-length.c
Determines the length of a message */

#include <stdio.h>

int main(void)
{
  char ch;
  int len = 0;

  printf("Enter a message: ");
  ch = getchar();
  while (ch != '\n') {
    len++;
    ch = getchar();
  }
  printf("Your message was \
         %d character(s) long.\n", len);

  return 0;
}
```

7.4 类型转换

- **隐式**转换
  - 操作数类型不同
  - 赋值运算
- 常用数据转换
  - 提升
  - 有符号数和无符号数混用
    - 可能出现错误
    - 建议不要用无符号数

```
char c;
short int s;
int i;
unsigned int u;
long int l;
unsigned long int ul;
float f;
double d;
long double ld;

i = i + c;     /* c is converted to int                */
i = i + s;     /* s is converted to int                */
u = u + i;     /* i is converted to unsigned int       */
l = l + u;     /* u is converted to long int           */
ul = ul + l;   /* l is converted to unsigned long int */
f = f + ul;    /* ul is converted to float             */
d = d + f;     /* f is converted to double             */
ld = ld + d;   /* d is converted to long double        */
```

7.4 类型转换

- 隐式转换
  - 赋值过程中的转换
    - 右边转换为左边的类型

```
char c;
int i;
float f;
double d;


i = c;   /* c is converted to int   */

f = i;   /* i is converted to float */
d = f;   /* f is converted to double */

c = 1000;     /*** WRONG ***/
i = 1.0e20;   /*** WRONG ***/
f = 1.0e100;  /*** WRONG ***/
```

```
int i;

i = 842.97;     /* i is now 842 */
i = -842.97;    /* i is now -842 */


f = 3.14159f;
```

7.4 类型转换

- 强制类型转换

（ *类型名* ） 表达式

- 例：求浮点数小数部分

```
float f, frac_part;

frac_part = f - (int) f;
```

- 两个整数相除，希望得到浮点数结果

```
float quotient;
int dividend, divisor;
quotient = (float) dividend / divisor;
```

- 防止溢出

```
long i;
int j = 1000;

i = (long) j * j;
```

7.5 类型定义

typedef

```
typedef float Dollars;
Dollars cash_in, cash_out;
float cash_in, cash_out;

typedef int Quantity;
Quantity q;

typedef long Quantity;
```

7.6 sizeof运算符

- 格式

  `sizeof ( type-name )`

- 值：无符号数，存储该类型名所需字节数
  - sizeof(i)/4
  - sizeof(i+j)/4