本小节内容:

数据类型
常量
变量(整型-浮点-字符)

1 数据类型

裁缝做衣服时需要用到化纤、纯棉、丝绸等不同类型的布料，那么程序员在编写程序时需要用到哪些数据类型呢？数据类型的分类如下图所示。

数据类型的分类

C 语言中有许多关键字，在后面的章节中将详细介绍这些关键字 **(不用去记)**，这里罗列它们的目的是让大家知道 C 语言中的关键字有哪些，**以避免命名变量时与关键字重名（完全不用担心，CLion 开发环境重名后会自动提醒大家的）.** 下表列出了 C 语言中的关键字。

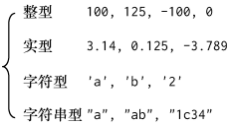
C 语言中的关键字

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

2 常量

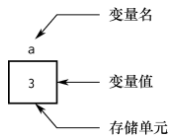
常量是指在程序运行过程中，其值不发生变化的量。常量又可分为**整型、实型（也称浮点型）、字符型和字符串型**，如右图所示。

整型常量、实型常量、字符型常量是在编译时可以直接编入代码段的常量；例如，在字符串"你好"中，双引号中间的汉字就是字符串型常量，无论双引号中间的内容是 ASCII 码字符，还是汉字或其他国家的文字等，都是字符串型常量（考研不会考汉字的字符串常量，无需掌握）。



常量的分类

3 变量



变量名、变量值和存储单元的关系

变量代表内存中具有特定属性的一个存储单元，它用来存放数据，即变量的值。**这些值在程序的执行过程中是可以改变的。**

变量名实际上以一个名字代表一个对应的存储单元地址。编译、链接程序时，由编译系统为**每个变量名分配对应的内存地址（就是空间）**。从变量中取值实际上是通过变量名找到内存中存储单元的地址，并从该存储单元中读取数据，如左图所示。

变量的命名规定如下：C 语言规定标识符只能由**字母、数字和下画线**三种字符组成，并且**第一个字符必须为字母或下画线**。例如，

```
sum, _total, month, Student_name, lotus_1_2_3, BASIC, li_ling
```

是正确的；而

```
M.D.John, ￥123,3D64, a>b
```

是错误的。

编译系统认为大写字母和小写字母是不同的字符，因此 C 语言要求对所有用到的变量做强制定义，即“先定义，后使用”。同时在选择变量名和其他标识符时，应尽量做到“**见名知意**”，即选择具有含义的英文单词（或其缩写）作为标识符。**注意，变量名不能与关键字同名!**

4 整型数据

4.1 符号常量

定义一个整型变量时要使用关键字 int。我们来看下面的例子。

```
#include <stdio.h>

#define PI 3+2
int main()
{
    int i=PI*2;
    printf("i=%d\n",i);
}
```

最终的输出结果是7，原因是符号常量 PI 是直接替换的效果，因此不可以写 PI=8。

4.2 整型变量

这里掌握 int i 足以应对初试，后面高级阶段会详细讲解不同类型整型变量，没有时间的同学可以不掌握高级，**变量 i 是 4 个字节**

5 浮点型数据

5.1 浮点型常量

表示浮点型常量的形式有两种，如下表所示，其中 e 代表 10 的幂次，幂次可正可负。

表示浮点型常量的两种形式

小数形式	0.123
指数形式	3e−3（为 3×10^{-3}，即 0.003）

注意：字母 e（或 E）之前必须有数字，且 e 后面的指数必须为整数。

正确示例：1e3、1.8e−3、−123e−6、−.1e−3。

错误示例：e3、2.1e3.5、.e3、e。

5.2 浮点型变量

通过 float f 来定义浮点变量，f 占用 4 个字节的空间

6 字符型数据

6.1 字符型常量

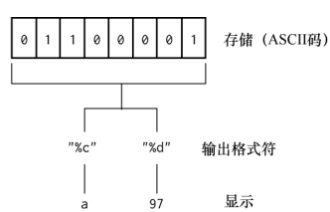
用单引号括起来的一个字符是字符型常量，且只能包含一个字符！例如，'a'、'A'、'1'、' '是正确的字符型常量，而'abc'、"a"、""是错误的字符型常量。下表中给出了各种转义字符及其作用。以“\”开头的特殊字符称为转义字符，转义字符用来表示回车、退格等功能键。

各种转义字符及其作用

转义字符	作　用
\n	换行
\b	退格
\\	反斜杠

6.2 字符数据在内存中的存储形式及其使用方法

字符型变量使用关键字 char 进行定义，**一个字符型变量占用 1 字节大小的空间。**一个字符常量存放到一个字符型变量中时，实际上并不是把该字符的字型放到内存中，而是把该字符的 ASCII 码值放到存储单元中，每个字符的 ASCII 码值详见附录 A。打印字符型变量时，如果以字符形式打印，那么计算机会到 ASCII 码表中查找字符型变量的 ASCII 码值，查到对应的字符后会显示对应的字符，



找 ASCII 码值并显示对应的字符

如右图所示。这样，字符型数据和整型数据之间就可以通用。字符型数据既可以以字符形式输出，又可以以整数形式输出，还可以通过运算获取想要的各种字符，请看下面例子。

```
#include <stdio.h>
int main()
{
    char c='A';
    printf("%c\n",c+32);
    printf("%d\n",c);
}
```

对于字符型变量，无论是赋 ASCII 码值还是赋字符，使用%c 打印输出时得到的都是字符，使用%d打印输出时得到的都是 ASCII 码值。将小写字母转换为大写字母时，由课件最后的 ASCII 码表发现小写字母与大写字母的差值为 32，因此将 c 减去 32 就可以得到大写字母 A。

7 字符串型常量

字符串型常量是由一对双引号括起来的字符序列。例如，"How do you do."、"CHINA"、"a"和"$123.45"是合法的字符串型常量，我们可用语句 printf("How do you do.")输出一个字符串。**但要注意的是，'a'是字符型常量，而"a"是字符串型常量，二者是不同的。**

例如，如果先用语句 char c 定义字符型变量 c，后令 c="a"或 c="CHINA"，那么这样的赋值都是非法的，原因是不可以将字符串型常量赋值给字符型变量。C 语言中没有定义字符串型变量的关键字，介绍字符数组时我们将详细讲解如何存放字符串。

C 语言规定，在每个字符串型常量的结尾加一个字符串结束标志，以便系统据此判断字符串是否结束。C 语言规定以字符'\0'作为字符串结束标志。

例如，字符串型常量"CHINA"在内存中的存储结果如下图所示，它占用的内存单元不是 5 个字符，而是 6 个字符，即大小为 6 字节，最后一个字符为'\0'。然而，在输出时不输出'\0'，因为'\0'无法显示。

C	H	I	N	A	\0

字符串型常量"CHINA"在内存中的存储结果

ASCII表

高四位 / 低四位		ASCII 码控制字符											ASCII 码打印字符																	
		0000						0001					0010		0011		01/0		0101		0110		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	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	横向指标	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	SOH		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				

注：表中的ASCII字符可以用"Alt + 小键盘上的数字键"方法输入