程序设计语言与方法(C语言)

第二章 数据表示与数据类型

认识一下

- 打个招呼

```
#include <stdio.h>
int main() /* 主函数 */
{
  /* 调用函数输出一个字符串 */
  printf("hello world!");
  printf("how are you!");
}
```

头文件

注释

语句

我就是一个函数 :）

数据

- 请罗列出一些数据
- 它们都是数值吗?
- 都是整数吗?
- 它们会变化吗?
- 它们容易记吗?
- 还会有其它的数据吗?
- 它们是独立的吗?
- 如何区分这些数据呢?
- 如何长期保存数据?

一些数据

- 10，20，3，1080
- -10，-512
- 3.14159，9.8
- -0.7，-1.99
- X, Y,Z，男,女
- 年、月(1-12)、日(1-31)
- 张三，李四
- 真、假
- 2014-10-9
- 19:00:07
- 2014-10-9 19:00:07
- 向量(1,3,4,5)
- 矩阵
- 地址（另一个地址）

分类数据

- 值域
  - 无限：整数、实数、正数、负数、日期……
  - 有限：月、日、星期、时、分、秒、……
- 符号

问题：计算机如何识别这些数据？

- 逻辑（布尔、二值）
- 同类数据序列（集合）
  - 字符序列、向量、矩阵、……
- 组合数据
- 构成完整的信息

计算机中的数据

- 二进制，也只有二进制数据
- 存储器
- 存储器那么大，数据在哪?
- 如何才能找到需要的数据?

二进制

- 进制
  - 十进制、二进制、八进制、十六进制
- 数值分类
  - 定点数
    - 整数、有符号数、无符号数
    - 定点小数
  - 浮点数（实数）

进制转换：数值的不同进制表示

编码（数据表示的唯一方法）

- 数值表示（整数）
  - 原码、反码
  - 补码（减法的加法实现）
- 用二进制数据表示其它类型的数据
  - ASCII码（字符的数值表示）
    - 标准字符集、扩展字符集

ASCII字符集

ASCII 字符代码表 一

高四位 / 低四位		ASCII非打印控制字符										ASCII 打印字符												
		0000					0001					0010		0011		0100		0101		0110		0111		
		0					1					2		3		4		5		6		7		
		+进制	字符	ctrl	代码	字符解释	+进制	字符	ctrl	代码	字符解释	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	ctrl
0000	0	0	BLANK NULL	^@	NUL	空	16	►	^P	DLE	数据链路转意	32		48	0	64	@	80	P	96	`	112	p	
0001	1	1	☺	^A	SOH	头标开始	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	反确认	37	%	53	5	69	E	85	U	101	e	117	u	
0110	6	6	♠	^F	ACK	确认	22	■	^V	SYN	同步空闲	38	&	54	6	70	F	86	V	102	f	118	v	
0111	7	7	•	^G	BEL	震铃	23	↨	^W	ETB	传输块结束	39	'	55	7	71	G	87	W	103	g	119	w	
1000	8	8	◘	^H	BS	退格	24	↑	^X	CAN	取消	40	(	56	8	72	H	88	X	104	h	120	x	
1001	9	9	○	^I	TAB	水平制表符	25	↓	^Y	EM	媒体结束	41	)	57	9	73	I	89	Y	105	i	121	y	
1010	A	10	◙	^J	LF	换行/新行	26	→	^Z	SUB	替换	42	*	58	:	74	J	90	Z	106	j	122	z	
1011	B	11	♂	^K	VT	竖直制表符	27	←	^[	ESC	转意	43	+	59	;	75	K	91	[	107	k	123	{	
1100	C	12	♀	^L	FF	换页/新页	28	∟	^\	FS	文件分隔符	44	,	60	<	76	L	92	\	108	l	124	|	
1101	D	13	♪	^M	CR	回车	29	↔	^]	GS	组分隔符	45	-	61	=	77	M	93	]	109	m	125	}	
1110	E	14	♫	^N	SO	移出	30	▲	^6	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	Δ	^Back space

注：表中的ASCII字符可以用：ALT + “小键盘上的数字键” 输入

其它字符编码

- unicode
- utf-8
- utf-16
- utf-32
- gbk2312
- ……

存储器

- 一堆盒子（数量有限的存储单元）
- 盒子（存储单元，可放置有限位的二进制数）
- 任何时刻一个盒子只能放一个数据（可覆盖性）
- 如何放数据入某个盒子里?
- 如何从给定的盒子里获取数据?
- 数据大了，怎么办?

存储单位

- 位（bit，一个二进制位，0/1）
- 字节（Byte，所谓的盒子，数据的基本存储单元）
  - 8bit，字节所包含的二进制位数，所谓的盒子大小
- 字（word，两个连续的字节）
- 双字（double word，两个连续的字）
- 四字（quad word，两个连续的双字）

数据访问

- 编码每个字节的位置
  - 给每个字节一个唯一的地址
- 字节地址从0开始，顺序编码
- 存储块（若干个连续的字节）
  - 字、双字、四字、……

问题

- 数据到底用几个字节存放?
- 存储块的大小如何确定?
- 如何快速找到存储块?
- 如何容易记住使用的是哪个存储块?

程序设计语言中的数据表示

以更自然的方式使用数据

- 数据类型
  - 设定存储块的大小（限制数据的值域）
- 命名存储块
  - 使数据容易使用
- 赋值
  - 变更存储块中的数据

数据类型

每种数据类型都有固定的名称！！

- 数据表示、存储和访问的处理方式
- 基本类型（简单数据类型）
  - 整型int：短整型、整型、长整型、无符号的（短整型、整型、长整型）
  - 实型：单精度实型float、双精度实型double、长双精度实型long double
  - 字符类型char、枚举类型enum
- 构造类型（复杂数据类型）
  - 数组、结构体、共用体
- 指针类型、无类型(void)

常量：直接使用数据

- 使用期间，数值不变的量
- 圆周率、重点加速度、身份证号、人的名字

- 直接常量
- 符号常量
- 使变量的值不可变化

直接常量

- 常数、幻数
  - 直接用值来表示的数据
- 数值常量（整数）
  - 有符号整数
    - 20 -30 0 -256l 1024L； l/L 表示整数是长整数（大数据的表示）
  - 无符号整数
    - 20 30lu 512LU； lu/Lu/lU/LU 表示无符号的长整数（大数据表示）

直接常量

- 数值常量（实数数）
  - 十进制小数：0.123 .5 -12.35 1.24f/1.24F 1.25L/1.25l
  - 指数：2e2 -2.0e2 -2.0e-2 3.45e-6 1.2e-8 2.5e-2f/2.5e-2F
- 字符
  - 'A' '0' ‘ ’
- 字符串
  - "ABC" "a02" "02a" " 02 ab c "

直接常量使用时的问题

- 导致程序的可读性变差：这个数据代表啥意思?
- 难以保证多次书写同一数据时的一致性；
- 常数需要变更时，工作量大且可能会有遗漏；

符号常量（宏常量）

- 如何保证多次使用的常量的一致性（都是一个值）？
- 符号化：给常量一个名称
  - #define PI 3.14159
- 使用名称来代替常量
  - PI
  - PI * r * r

符号常量示例

```c
#include <stdio.h>

#define PI 3.14159

int main()
{
    int r = 2;
    double s;
    s = PI * r * r;
    printf("%f\n", s);
    return 0;
}
```

宏定义

PI 是宏名

宏替换

```c
#include <stdio.h>

int main()
{
    int r = 2;
    double s;
    s = 3.14159 * r * r;
    printf("%f\n", s);
    return 0;
}
```

命名存储块（变量）

- 存储块的符号化表示
- 标识符(Identifier)：用户自定义的一个名称，不同于关键字（Keyword）
  - 英文字母、下划线、数字
    - 必须以下划线或字母开始
    - 不允许使用关键字作为标识符
    - 可以包含多个字符，但通常会有最大长度的限制
    - 区分大小写，例如：area、Area、AREA、……

变量定义

- C语言中的数据（值可变化）表示
  - 定义待存储数据的类型（确定存储块的大小）
  - 定义存储块（变量）的名称（易于使用）
- 示例
  - int x、short int x1、unsigned int _xx_d_e3
  - double PI、long double y

如何确定变量的数据类型和名称?
值域包容、含义尽可能要明了

变量赋值

- 将数据放入既定的存储块
- 赋值语句

- int x;
- x = 23; /* 只能赋值符合值域要求的数据*/

定义有初始值的变量

- 变量、有初始值
- 示例

- **int x = 23;**
- **double PI=3.14159;**
- **char ch = ‘A’;**

同时定义多个变量

- 必须是同类型的一组变量
- 逗号分隔
- 示例
  - int x, y, z;
  - int x = 10, y = 20, z = 30;

简单的屏幕输出

```
#include <stdio.h> /* 头文件 */
int main()   /* 主函数，必须是，也只能是这样 */
{
  int a = 12; /* 定义整型变量a并对其初始化 */
  float b = 2.5; /* 定义实型变量b并对其初始化 */
  char c = 'A'; /* 定义实型变量c并对其初始化 */
  printf("a = %d\n", a); /* 按整型格式输出变量a的值 */
  printf("b = %f\n", b); /* 按实型格式输出变量b的值 */
  printf("c = %c\n", c; /* 按字符型格式输出变量c的值 */
  printf("End of program \n"); /* 输出一个字符串 */
 }
```

const 常量：有类型的常量

- 符号/宏常量的问题
  - 没有数据类型，编译器也不会对其做类型检查，只是进行简单的符号替换，易出错

- const 常量
  - 其值固定不变的变量
    - const double PI=3.14159;
      - 2＊PI＊2：编译器会报类型检查的警告信息（编译器相关）